UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386

HOCHSCHULE HEILBRONN
TECHNIK WIRTSCHAFT INFORMATIK

Entwicklung und Realisierung von BCI-Experimenten für die Ausbildung im Bereich der medizinischen Signalverarbeitung auf Basis des Buffer-BCI-Frameworks

Bachelor-Thesis

zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.) im Studiengang
Medizinische Informatik

vorgelegt von

Beatrice Baumann

183460

am

25. September 2015

Referent: Prof. Dr. Rolf Bendl

Korreferent: Dr. Christoph Maier

Betreuer: Dr. Christoph Maier

Abstract

Eine Anwendung nur mit Gedanken steuern? — Das geht! Mit Hilfe von Brain-Computer-Interfaces (BCIs). Heutzutage gibt es viele BCI-Frameworks, bei denen die unterschiedlichen Konzepte wie zum Beispiel P300-Speller, Imagined Movement oder SSVEP untersucht und angewendet werden können.

Für diese Thesis wird das Framework Buffer-BCI verwendet. Es wurde untersucht, ob sich dieses für eine Aufgabe im Praktikum Medizinische Signal- und Bildverarbeitung (PSB) des Studiengangs Medizinische Informatik eignet. An mehreren Probanden wurde getestet, ob Reproduzierbarkeit und Erkennungsgenauigkeit hoch genug sind, um auch bei vielen unterschiedlichen Menschen vergleichbar funktionieren und sinnvolle Ergebnisse liefern zu können. Auch hinsichtlich zeitlicher Latenzen, der prinzipiellen Architektur des Frameworks und der Kommunikationskette für Ereignisse wurde Buffer-BCI untersucht.

Ein Ergebnis dieser Thesis ist die Beschreibung der Architektur und der Kommunikationskette von Buffer-BCI. Ein weiteres Ergebnis ist eine Aufgabenstellung für das PSB bezüglich einer P300-Speller-Anwendung auf Basis des Buffer-BCI-Frameworks. Die Aufgabenstellung kann noch um Imagined Movement- oder SSVEP-Experimente erweitert werden, was jedoch nicht mehr Teil dieser Thesis ist.

Danksagung

Zuallererst möchte ich meiner Familie danken, die mir mein Leben lang geholfen hat, wo sie nur konnte. Sie hat immer an mich geglaubt und mich unterstützt, wo es nur ging. Danke dafür.

Ich danke Herrn Maier, für das engagierte und freundliche Betreuen während der Thesis. Vielen Dank, für das beantworten meiner Fragen in diesem komplexen Themenbereich.

Ein weiterer Dank gilt Herrn Bendl, der mich auf dieses Thema aufmerksam machte und ebenfalls betreute.

Bei Peter Seitz möchte ich mich für seine Unterstützung beim Erstellen der Grafiken und für das Korrekturlesen bedanken.

Weiterhin danke ich allen anderen Personen, die mir zur Seite standen, mich ermutigt haben und mir geholfen haben.

Inhaltsverzeichnis

Tabellenverzeichnis

List of Listings

Abbildungsverzeichnis

Abkürzungsverzeichnis

BCI	Brain-Computer-Interface
BMI	Brain-Machine-Interface
BS	Betriebssystem
EEG	Elektroenzephalogramm
FUSBI	Glasfaser zu USB Schnittstelle
GND	Groundelektrode
GUI	Graphical User Interface
IM	Imagined Movement
PSB	Praktikum Medizinische Signal- und Bildverarbeitung
PTB	Psychophysics Toolbox
REF	Referenzelektrode
SSVEP	Steady State Visual Evoked Potential
SVN	Subversion

1 Einleitung

1.1 Gegenstand und Bedeutung

Nachdem Hans Berger im Jahr 1929 erstmals das Elektroenzephalogramm (EEG)[1] vorstellte, wurde der Grundstein für Brain-Computer-Interfaces (BCIs) gelegt. Ein BCI ist eine Schnittstelle zwischen Hirn und Computer, die es ermöglicht Anwendungen nur mit Gedanken steuern zu können. Das BCI spielt heutzutage in der Forschung eine große Rolle. Viele Forscher erhoffen sich, mit Hilfe von BCIs/Brain-Machine-Interfaces (BMIs), gelähmte Personen oder Menschen, die von neuronalen Erkrankungen betroffen sind, im Alltag unterstützen zu können. Wie weit die Forschung mittlerweile fortgeschritten ist, wird in Abschnitt 2.4 aufgezeigt. Da sich in diesem Bereich der Medizin bzw. der Medizintechnik auch in Zukunft noch viele Forschungsmöglichkeiten ergeben werden, ist es speziell für Medizininformatiker von Vorteil, sich schon früh mit diesem auseinander zu setzen. Darauf stützt sich meine Motivation.

1.2 Motivation

Im Praktikum Medizinische Signal- und Bildverarbeitung (PSB) des Studiengangs Medizinische Informatik gibt es bezüglich des EEGs bisher nur eine Aufgabe. In dieser werden hauptsächlich die diagnostischen Eigenschaften eines EEGs behandelt. Es werden Signale aufgezeichnet und analysiert. Jedoch kommt es zu keiner weitergehenden Anwendung mit EEG-Signalen, wie zum Beispiel Steuerungsaufgaben. Da dem Labor mittlerweile neue Hardware zur Verfügung steht, die weitergehende Anwendungen ermöglicht, soll sich dies mit Hilfe dieser Thesis ändern. Die Studierenden sollen die verschiedenen BCI-Konzepte kennen lernen, die mit einem nichtinvasiv registrierten EEG bedient werden können. Um den Studierenden eine Einführung in die verschiedenen BCI-Konzepte geben zu können, soll eine Aufgabe für das PSB realisiert werden.

1.3 Ziele

Die Ziele dieser Bachelorthesis sind in der nachfolgenden Aufzählung erläutert:

1. Das Framework Buffer-BCI auf seine Eignung für eine Aufgabe im PSB untersuchen
    - Wie sicher und reproduzierbar sind die einzelnen Anwendungen?
    - Wie groß sind verschiedene zeitliche Latenzen?
    - Wie groß ist der zeitliche Aufwand (Aufbau der Hardware, Ausführen der Anwendung)
    - Wie aufwändig ist es, Buffer-BCI um eigene Funktionen zu erweitern?

[1] erstmals Veröffentlichter Bericht über das EEG[1]

2. Die prinzipielle Architektur des Systems und der Kommunikationskette für EEG-Signale und Ereignisse dokumentieren
    - Wie hängen die einzelnen Komponenten, die Buffer-BCI mit sich bringt, zusammen?
    - Wie und worüber kommunizieren sie miteinander?
    - Wie werden die Hirnsignale erfasst?
3. Ein BCI-Experiment realisieren, welches eine Einführung in die verschiedenen BCI-Konzepte gibt
    - Untersuchen, wie relevante Signalsegmente und Zwischenergebnisse im Praktikumsbetrieb verfügbar gemacht und visualisiert werden können.
    - Aufgabenstellung für eine Laborübung im Rahmen des PSB erstellen.

1.4 Vorgehensweise und Struktur der Arbeit

Zu Beginn der Thesis wurde viel mit Buffer-BCI experimentiert und getestet (Kapitel 5), um einen Einstieg in die Materie zu bekommen und sich mit der Hardware (Abschnitt 2.3) vertraut zu machen. Um die Synchronizität von Signalaufnahme einerseits und der Signalisierung von Ereignissen andererseits zu ermitteln, wurden verschiedene Latenzzeiten gemessen. Dies ist wichtig, da die Grundidee von Buffer-BCI das zeitliche Koppeln von (Software-) Ereignissen mit einem Hardware-TriggerSignal ist und es zwischen diesen zu keinem größeren Versatz kommen sollte. Auch die Architektur von Buffer-BCI und die Kommunikationskette wurden ermittelt. Um relevante Signalabschnitte für das PSB verfügbar zu machen, wurde Buffer-BCI um eine Anzeige der Reizantworten ergänzt. Eine Anforderungsanalyse (Kapitel 3) diente als Grundlage für die Untersuchung von Buffer-BCI (Kapitel 4), die eigenen praktischen Experimente (Kapitel 5) und für den Praktikumsversuch (Kapitel 6). Mit der Erstellung einer Aufgabenstellung für das PSB wurde die Thesis abgeschlossen.

2 Grundlagen

In diesem Kapitel werden grundlegende Begriffe, die für das Verständnis eines BCI gebraucht werden, erklärt. Des Weiteren wird die verwendete Soft- und Hardware sowie der aktuelle Stand der Forschung vorgestellt.

2.1 Grundlagen von EEG und BCIs

In diesem Abschnitt werden die Grundlagen erklärt, die man braucht um BCIs zu verstehen. Dazu gehört das EEG sowie die verschiedenen Ansätze eines BCIs.

2.1.1 Elektroenzephalogramm

Das Elektroenzephalogramm (EEG) ist ein elektronisch registriertes Signal, welches durch die Elektroenzephalographie gewonnen wird. Die Elektroenzephalographie ist eine diagnostische Methode zur Registrierung von Potentialschwankungen des Gehirns. Diese entstehen durch die Aktivität von Nervenzellen, welche im Bereich der Hirnrinde auftreten. Von auf der Kopfhaut angebrachten Elektroden werden diese erfasst, verstärkt und ohne Unterbrechung aufgezeichnet.[2] Die Signale müssen verstärkt werden, da es sich hierbei um sehr kleine Potentiale (von nur wenigen Mykrovolt) handelt. Um sicherstellen zu können, dass die Elektroden bei jedem Menschen vergleichbar angebracht werden können, gibt es das sogenannte 10-20-System. Bei diesem System wird der Schädelknochen vom Nasion (Ansatz der Nasenwurzel) bis zum Inion (Knochenvorsprung in der Mitte des Hinterhauptbeins) vermessen. Der Wert dieser Strecke wird als 100% angenommen. Diese wird vom Nasion in Richtung Inion in 10%, viermal 20% und nochmal 10% unterteilt. Das gleiche Verfahren verwendet man auch zwischen den beiden präaurikulären Punkten (bei den Ohren liegend). Von diesen Koordinaten ausgehend, werden die Elektroden auf der Kopfhaut angebracht.[3] Dieses System wird in Abbildung 2.1 verdeutlicht.

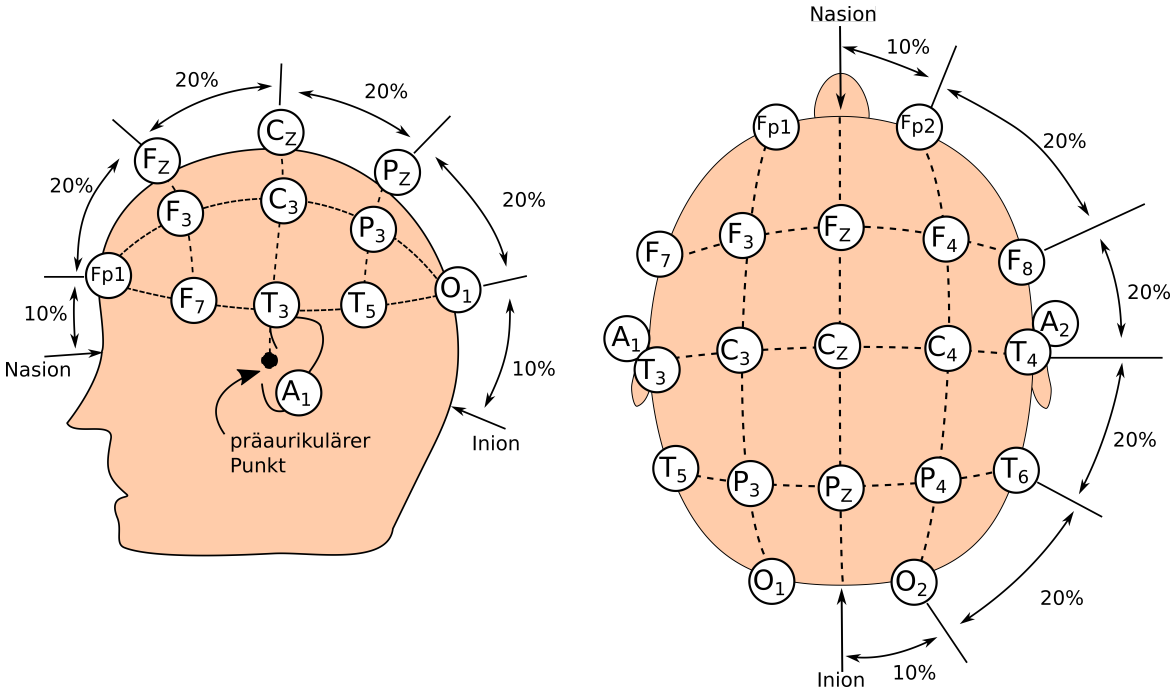


Abbildung 2.1: 10-20-System

2.1.2 Brain-Computer-Interface

(Syn: Brain-Machine-Interface (BMI)) Ein Brain-Computer-Interface (BCI) beschreibt eine direkte Schnittstelle zwischen Gehirn und Computer. Von diesen Schnittstellen gibt es sowohl invasive als auch nichtinvasive Varianten. Bei dem invasiven Ansatz werden Elektroden in das Gehirn implantiert, wohingegen bei der nichtinvasiven Variante lediglich das EEG aufgezeichnet wird. Um beim BCI Reize zu generieren, gibt es verschiedene Ansätze, die in den Abschnitten 2.1.2.1 und 2.1.2.2 erläutert werden. Beim BCI entsprechen die Reize einer Präsentation verschiedener Optionen, wobei der seltenere Reiz der "richtigen" Option entspricht. Damit ist das Problem des BCI im Fall der P300 die Feststellung des Vorliegens einer P300-Welle als Reaktion auf einen präsentierten Reiz. Diese Entscheidung muss in Quasi-Echtzeit getroffen werden.

2.1.2.1 Oddball-Paradigma

Das Oddball-Paradigma beschreibt ein Verfahren, bei dem Standardreize (Buchstabe leuchtet nicht) und Zielreize (Buchstabe leuchtet)[1] präsentiert werden. Ziel ist es, sich auf die Zielreize, die gegenüber den Standardreizen deutlich seltener erscheinen, zu konzentrieren.[4]

[1] jeweils bezogen auf die matrixSpeller-Anwendung

2.1.2.2 nichtinvasive Ansätze eines BCIs

Im folgenden Abschnitt werden die nichtinvasiven Ansätze eines BCIs vorgestellt. Hierzu zählen die P300, das Steady State Visual Evoked Potential (SSVEP) sowie das Imagined Movement (IM).

Klassisch ist die P300-Speller-Anwendung, bei der sich der Benutzer auf einen Buchstaben konzentriert. Das BCI muss mit Hilfe der P300 die Entscheidung treffen, welcher Buchstabe es war, den der Benutzer sich angesehen hat.

1. **P300** Die P300 ist eine positive Welle im EEG (siehe Abbildung 2.2), die als Antwort auf einen spezifischen Reiz gemessen werden kann und ca. 300 Millisekunden nach Reizbeginn auftritt. Sie lässt sich in die P3a und die P3b unterteilen, wobei die P3b der P300 entspricht. Es handelt sich bei der P300 grundsätzlich um ein reizsynchron gemitteltes evoziertes Potential, wobei es zwei verschiedene Reizklassen gibt von welchen der eine seltener ist, wie bereits in Abschnitt 2.1.2.1 erläutert. Dieser seltenere Reiz wird von der P300-Welle begleitet, der deutlich häufigere Reiz nicht.

   - **P300 oder P3b**
     Über den mittleren parietalen Elektroden (Pz) hat sie ihr Maximum. Seitlich davon fällt sie symmetrisch ab. Am einfachsten lässt sich eine P300 mit Hilfe des Oddball-Paradigma (Abschnitt 2.1.2.1) erzeugen.
   - **P3a**
     Wird in das Oddball-Paradigma ein dritter (auch seltener) Reiz aufgenommen, erscheint im EEG eine zusätzliche Welle — die P3a. Sie unterscheidet sich von der P3b durch eine kürzere Latenz. [5]

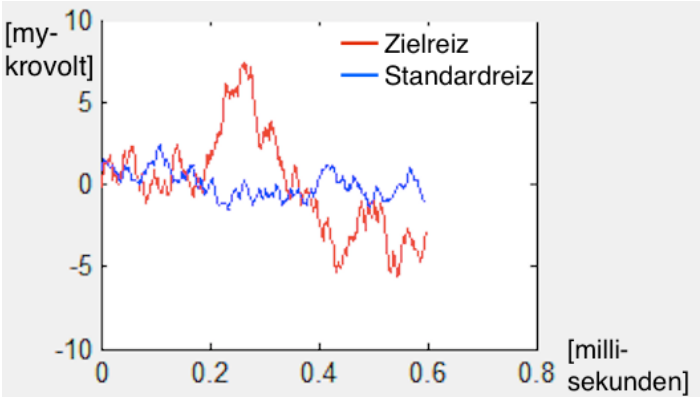


Abbildung 2.2: Selbst aufgenommene P300-Welle (rot)

2. **Steady State Visual Evoked Potential (SSVEP)** SSVEP sind Hirnsignale, die als Reaktion auf visuelle Stimulation auftreten. Diese Signale können von jedem sich wiederholenden Blinklicht ausgelöst werden. Das Anzeigen eines solchen Blinklichtes mit einer bestimmten Frequenz stimuliert die Sehnerven, was dazu führt, dass diese die Frequenz an das Gehirn weiterleiten. Im EEG erscheinen neben der ursprünglichen Frequenz auch die Vielfachen von dieser. Das Blinklicht wird im Falle einer SSVEP-Anwendung bei Buffer-BCI auf dem Computerbildschirm durch 4 Felder, die mit unterschiedlicher Frequenz blinken, realisiert. Diese sind gleichzeitig sichtbar und die Entscheidung, welches Feld der Benutzer angesehen hat, wird durch eine Frequenzanalyse des EEG getroffen. Diese Variante gilt als relativ robust.
3. **Imagined Movement (IM)** (deutsch: vorgestellte Bewegung) Der Ursprung willkürlicher Bewegungen ist der Motorcortex. Dieser ist so organisiert, dass bestimmte Bereiche des Motorcortex die Bewegung bestimmter Körperregionen kontrollieren. Dabei lassen sich über dem entsprechenden Bereich des Motorcortex Aktivierungen messen, wenn man sich die Bewegung des zugehörigen Körperteils nur vorstellt (der sogenannte mu-Rhythmus). Ziel des IM-BCIs ist es, diesen zu identifizieren.

2.1.3 Elektroden-Impedanz

Die Impedanz beschreibt in diesem Fall den Widerstand zwischen Kopfhaut und Elektrode. Dieser sollte möglichst gering sein (ca. 25 kΩ) um eine gute Signalaufzeichnung zu gewährleisten. Daher wird Elektrolyt Gel auf die Kopfhaut aufgespritzt.

2.2 Software

2.2.1 FieldTrip-Buffer

Der FieldTrip-Buffer ist die Grundlage für die Kommunikation des gesamten Buffer-BCI-Systems.

Er ist der zentrale Kommunikations-Verteiler und erlaubt es, Signale und Ereignisse übers Netz für viele Konsumenten verfügbar zu machen. Die Buffer-Kommunikation basiert auf TCP/IP-Sockets und ist damit von Plattform und Programmiersprache unabhängig. Wichtig ist dabei auch das Konzept der Entkopplung von Signal"produzent" und Signal"konsument". Der Buffer hält die Daten für eine bestimmte Dauer verfügbar (z.B. jeweils die letzten 10 Minuten). Somit geht nichts verloren wenn der Konsument mal nicht gleich die Daten vom Produzenten (A/D-Wandler) abholen kann weil er gerade anderweitig beschäftigt ist. Theoretisch ist der Buffer eine isolierte (Server-) Komponente, praktisch erfolgt die Realisierung jedoch oft in Kombination mit einer Client-Komponente, die spezifische Datenakquisitions-Hardware anspricht und die

Daten gleich an den Fieldtrip-Bufferserver schickt. Für die im Labor verfügbaren TMSi-Verstärker gab es das Programm tmsi2ft.exe, welches jedoch auf alten TMSi-Treibern basierte, die nur auf 32-Bit Windows Systemen lauffähig war.

2.2.2 tmsi2ft

Die neuen, 64-Bit tauglichen Treiber von TMSi besitzen eine geänderte Programmierschnittstelle, die es erlaubt, alle Kommunikationskanäle der TMSi-Verstärker (FUSBI, Bluetooth, WiFi, ...) einheitlich anzusprechen. Daher musste das alte tmsi2ft, dessen Quellcode auf der Fieldtrip-Website verfügbar ist, dahingehend umgeschrieben werden, dass es die neuen Treiber verwendet. tmsi2ft kann auf Wunsch eine eigene Instanz des FieldTrip-Buffers starten.

2.2.3 tmsi_refa_config.txt

Bei dieser Datei handelt es sich um eine Konfigurationsdatei. In dieser Datei können Elektroden ausgewählt werden, indem man das #-Zeichen vor der jeweiligen Elektrode löscht, wie in Abbildung 2.3 zu sehen. Alle Elektroden mit #-Zeichen sind auskommentiert und werden somit nicht berücksichtigt.

```
#
1=Fp1
2=Fpz
3=Fp2
4=F7
5=F3
6=Fz
7=F4
8=F8
#9=FC5
#10=FC1
#11=FC2
```

Abbildung 2.3: Ausschnitt aus der Konfigurationsdatei: Elektroden 1 bis 8 sind ausgewählt

2.2.4 Psychophysics Toolbox

Die Psychophysics Toolbox (PTB) erweitert "MATLAB um Funktionen zur präzisen Reizpräsentation, zur Antwortmessung, sowie um eine Sammlung von Hilfsroutinen, die oft in der Experimentalprogrammierung gebraucht werden."[6] Die PTB benötigt ein 64-Bit Betriebssystem.

2.2.5 MATLAB

"MATLAB ist eine hochentwickelte Sprache und interaktive Umgebung, die von Millionen Entwicklern und Wissenschaftlern weltweit genutzt wird."[2][7] Für das Zusammenspiel mit Buffer-BCI wird die 64-bit Version MATLAB R2013b verwendet. Es wird eine bereits ältere Version verwendet, weil es in den neueren Versionen Änderungen z.B. an der Grafikschnittstelle gibt, die in Buffer-BCI noch nicht angepasst wurden. Die 64-Bit Version wird wegen der Psychtoolbox benötigt.

2.3 Hardware

Im folgenden Abschnitt werden die einzelnen Geräte, die im Labor verfügbar sind und in dieser Thesis verwendet werden kurz erklärt.

2.3.1 actiCap-System

Das actiCap-System ist ein System zur Signalaufzeichnung von Gehirnwellen. Zu diesem System gehören die actiCap, Elektroden, Controlbox, Splitterbox und Adapter zu verschiedenen Verstärkern. Die einzelnen Komponenten des Systems werden im folgenden Abschnitt beschrieben.

Abbildung 2.4: Das actiCap-System mit allen Komponenten

2.3.1.1 actiCap

Die actiCap wird wie eine Badekappe angezogen. An ihr sind Halterungen befestigt. In diese werden die Elektroden eingesetzt und mit Elektrolyt Gel befüllt, um eine bessere Verbindung zur Kopfhaut, beziehungsweise eine niedrige Impedanz, zu bekommen. Die Halterungen sind nach dem 10-20-System auf der actiCap angebracht.

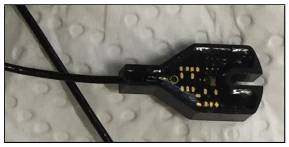
[2]Eigenwerbung von MATLAB

Abbildung 2.5: actiCap

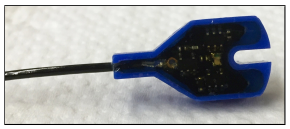
2.3.1.2 Elektroden

Bei den Elektroden handelt es sich um aktive Elektroden. Das bedeutet zum einen, dass direkt in der Elektrode bereits ein Verstärker integriert ist, um eine Verbesserung des Signalrauschens zu erreichen, zum anderen kann bei diesen Elektroden eine Impedanzüberwachung realisiert werden.[8]

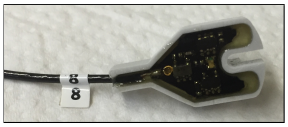
- Elektroden 1 - 32 (Datenelektroden)
  Die Elektroden werden über die Splitterbox mit der ControlBox verbunden.
- Groundelektrode (GND) und Referenzelektrode (REF)
  Die GND und REF müssen an der ControlBox angeschlossen sein, da sonst das Elektrodensystem nicht funktioniert.



(a) Groundelektrode (GND)



(b) Referenzelektrode (REF)



(c) Datenelektroden (insgesamt 32 Stück)

Abbildung 2.6: Die 3 verschiedenen Elektrodenarten

2.3.1.3 Controlbox

(Im Labor: Controlbox Version II) An der Controlbox sind die Splitterbox und der actiCap-Adapter angeschlossen. Die Splitterbox wird an den Anschluss Ch. 1-32 Splitterbox angeschlossen, der actiCap-Adapter an den Anschluss Ch. 1-32 Amplifier. Außerdem werden mit ihr auch die GND und REF verbunden.

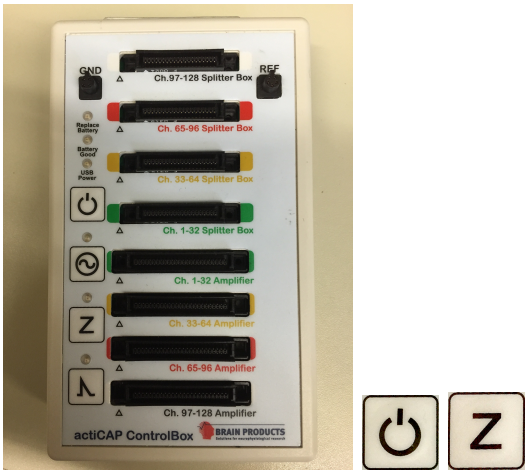


Abbildung 2.7: Controlbox, Tasten für Akquisitionsmodus und Impedanzmodus

- **Akquisitionsmodus** In diesem Modus können die EEG-Signale zum Verstärker übertragen und aufgezeichnet werden.
- **Impedanzmodus** In diesem Modus können die Impedanzen der Elektroden gemessen werden. In den Elektroden sind LEDs eingebaut, die folgende Impedanzen anzeigen:
  - grüne LED: Impedanz unter 25 kΩ
  - gelbe LED: Impedanz zwischen 25 und 60 kΩ
  - rote LED: Impedanz größer als 60 kΩ

  Um eine hervorragende Datenqualität während der EEG-Aufzeichnung zu bekommen, ist bei einer Verwendung von aktiven Elektroden eine Impedanz von 25 kΩ vollkommen ausreichend.

2.3.1.4 Splitterbox

An der Splitterbox sind alle Elektroden angeschlossen (Ausnahme: GND und REF). Die Splitterbox wird mit der Controlbox verbunden.

Abbildung 2.8: Splitterbox

2.3.1.5 Adapter

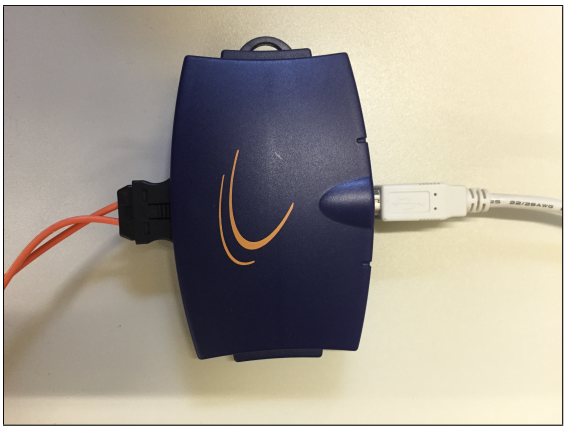
Es gibt verschiedene Adapter für verschiedene Verstärker. Im Labor ist ein Adapter für den TMSi-Verstärker vorhanden. Dieser ist nötig, damit der TMSI-Verstärker mit dem actiCap-System interagieren kann. Bei diesem Adapter kann jede Elektrode einzeln ein- oder ausgeschaltet werden.

Abbildung 2.9: actiCap-Adapter

2.3.2 TMSi-Verstärker

(Im Labor: TMSi Refa 8-32e4b4a) Der TMSi-Verstärker wird mit einem Glasfaserkabel über eine Glasfaser zu USB Schnittstelle (FUSBI) direkt mit dem Computer verbunden. Unter anderem besitzt der Verstärker 32 unipolare EEG-Kanäle, 4 weitere bipolare Eingänge sowie einen 8-Bit Trigger-Eingang. Die maximale Abtastrate des Verstärkers beträgt 2048 Hertz. Er wird über den actiCap-Adapter in das actiCap-System integriert.

(a) TMSi-Verstärker



(b) FUSBI

Abbildung 2.10: Hardware bezüglich Verstärker

2.3.3 Computer

Da die PTB, welche für manche Anwendungen benötigt wird, nur mit einem 64-Bit MATLAB funktioniert und TMSi nur Treiber für Windows zur Verfügung stellt, wird ein Computer mit einem 64-Bit Windows Betriebssystem (BS) gebraucht.

2.4 Wie weit ist die Forschung → Ein paar Forschungsergebnisse

Da es sehr viele Ergebnisse gibt, sind in der folgenden Aufzählung nur ein paar Beispiele zu sehen, die bis heute in der BCI-/BMI-Forschung erreicht wurden.

- 29. Juni 2009: Forscher von Toyota und Riken[3] entwickelten einen Rollstuhl, der mit Gedankenkraft nach links, rechts und vorwärts gesteuert werden kann.[9]
- 03. April 2013: Es konnte nachgewiesen werden, dass sich das Sehsystem des Menschen dazu eignet ein schnelles, zuverlässiges BCI zu konstruieren. [10]
- 27. August 2014: Kommunikation über Hirnströme kann schneller und zuverlässiger sein als über Unterstützungstechnologien (Bspw. ein Knopf), die auf Muskelaktivität basieren. [11]
- Am 17. August 2015 haben Mitarbeiter der Korea University und der Technischen Universität Berlin ein Exoskelett vorgestellt, welches mittels SSVEP gesteuert wird. [12]

[3]Forschungsinstitut im Bereich der Naturwissenschaften in Japan

3 Anforderungsanalyse

In diesem Kapitel wird erläutert, welche Anforderungen an die Thesis gestellt werden. Die Anforderungen leiten sich von den Zielen aus Abschnitt 1.3 ab.

3.1 Das Framework Buffer-BCI auf seine Eignung für eine Aufgabe im PSB untersuchen

Das Framework soll hinsichtlich der Anwendungen, den zeitlichen Latenzen, dem Zeitaufwand und der Erweiterbarkeit untersucht werden.

- **Anwendungen**
  Es soll überprüft werden, ob die Anwendungen funktionieren und die Ergebnisse auch über viele verschiedene Personen hinweg reproduzierbar sind.
- **Latenzen**
  Es soll der zeitliche Versatz zwischen Signalaufzeichnung und Trigger-Ereignissen untersucht werden.
- **Zeitaufwand**
  Der Zeitaufwand zum Durchführen einer Aufgabe sollte im Rahmen einer typischen Praktikumsaufgabe liegen. Um dies zu überprüfen, soll gemessen werden, wie lange es dauert die matrixSpeller-Anwendung auszuführen, und ob es Möglichkeiten gibt, diese Zeit zu verkürzen.
- **Erweiterbarkeit**
  Das Framework Buffer-BCI soll um eigene Funktionen, die relevante Signalsegmente anzeigen, erweitert werden.

3.2 Die Architektur des Systems und der Kommunikationskette für EEG-Signale und Ereignisse dokumentieren

Es soll dargestellt werden, wie Buffer-BCI intern abläuft. Hierzu soll untersucht werden, wie einzelne Komponenten des Frameworks miteinander in Zusammenhang stehen. Des Weiteren soll erläutert werden, wie und worüber die Komponenten miteinander kommunizieren.

3.3 BCI-Experiment realisieren

Es soll eine Versuchsanleitung mit einem BCI-Experiment für das PSB erstellt werden. Hierfür soll untersucht werden, wie relevante Signalsegmente und Zwischenergebnisse im Praktikumsbetrieb verfügbar gemacht und visualisiert werden können.

4 Untersuchung von Buffer-BCI

4.1 Vorgehen

Nach der Installation von Buffer-BCI wurde am Anfang viel experimentiert und getestet, um sich mit dem System vertraut zu machen. Es wurden viele wichtige Erfahrungen bezüglich Systemvoraussetzungen und Installation gemacht, die im Folgenden erläutert werden. Außerdem wurde die Architektur und Kommunikationskette von Buffer-BCI ermittelt. Des Weiteren wird beispielhaft an der P300-Speller-Anwendung gezeigt, wie man eine Anwendung von Buffer-BCI startet.

4.2 Systemvoraussetzungen

In diesem Abschnitt wird erläutert, welche Software ein Benutzer installieren müsste, um mit dem System arbeiten zu können. Es wird unterschieden zwischen Systemvoraussetzungen bei Anwendungen mit der PTB und den Systemvoraussetzungen bei Anwendungen ohne PTB (siehe Tabelle 4.1).

	Anwendungen **mit** PTB	Anwendungen **ohne** PTB
Betriebssystem (BS)	64-Bit Windows	32-Bit Windows
MATLAB	64-Bit MATLAB	32-Bit MATLAB
PTB	wird benötigt	wird nicht benötigt

Tabelle 4.1: Systemvoraussetzungen

Bei Anwendungen **mit** der PTB (4.4.1.2) wird ein 64-Bit BS benötigt, da die PTB nur unter diesem funktioniert. Es muss sich dabei um ein Windows BS handeln, da der im Labor verfügbare TMSi-Verstärker nur Treiber für Windows bietet. Auch MATLAB muss eine 64-Bit Version sein. Und selbstverständlich wird die PTB benötigt.

Bei Anwendungen **ohne** PTB (4.4.1.1)genügt ein 32-Bit BS. Jedoch muss es sich auch hierbei um ein Windows BS handeln, wegen dem TMSi-Verstärker. Ein 32-Bit MATLAB ist in diesem Fall ausreichend.

4.3 Installation

In diesem Abschnitt wird kurz erläutert, wie das Framework Buffer-BCI und die PTB installiert werden.

4.3.1 Buffer-BCI

Das Framework Buffer-BCI wird unter diesem Link `https://github.com/jadref/buffer_bci` aus dem Internet heruntergeladen und ist somit "installiert".

4.3.2 Psychophysics Toolbox (PTB)

Wenn noch kein Subversion (SVN) auf dem Computer vorhanden ist, muss es beispielsweise von diesem Link `https://sliksvn.com/download/` heruntergeladen und installiert werden. Für die Installation der PTB muss man die unter diesem Link `https://raw.githubusercontent.com/Psychtoolbox-3/Psychtoolbox-3/master/Psychtoolbox/DownloadPsychtoolbox.m` verfügbare Funktion kopieren und auf dem Desktop speichern. Dann muss auf der C-Festplatte ein Ordner "toolbox" kreiert werden. Die gespeicherte Funktion muss vom Desktop in den toolbox-Ordner eingefügt werden. Außerdem müssen die Microsoft Runtime Libraries for MSVC 2010 installiert sein. Nun muss MATLAB als Administrator ausgeführt werden. In der Kommandozeile von MATLAB muss nun folgendes eingegeben werden:

» cd C:\toolbox
» DownloadPsychtoolbox('C:\toolbox')

Dies kann nun einige Zeit (∼ 10 min) in Anspruch nehmen. Sollte es zu Schwierigkeiten während der Installation kommen, kann man sich unter diesem Link `http://psychtoolbox.org/download/#Windows` eine detailliertere Installationsanleitung anschauen.

4.4 Das Framework Buffer-BCI

Buffer-BCI ist ein platformunabhängiges Framework um BCI-Anwendungen zu entwickeln. Es basiert auf einer Client-Server-Architektur. Mehrere Clients senden und bekommen Events von einem zentralen Daten- und Eventserver. Dieser basiert auf dem FieldTrip-Buffer (2.2.1).[13]

4.4.1 Anwendungen und unterstütze BCI-Paradigmen

Im folgenden Abschnitt werden die Anwendungen und jeweils welche BCI-Paradigmen diese unterstützen vorgestellt. Sie werden unterteilt in Anwendungen, welche die PTB verwenden und in die Anwendungen, die sie nicht verwenden.

4.4.1.1 Anwendungen ohne PTB

In dieser Aufzählung sind alle Anwendungen von Buffer-BCI, die ohne die PTB auskommen, aufgezählt.

1. **P300**
   Die folgenden Anwendungen unterstützen das P300-Paradigma.

- **matrixSpeller** In dieser Anwendung wird eine Buchstabenmatrix dargestellt. In dieser Matrix leuchten nacheinander Zeilen und Spalten auf. Der Benutzer konzentriert sich auf einen Buchstaben. Er sollte möglichst nur diesen betrachten. Immer, wenn dieser Buchstabe aufleuchtet, kommt es zu einer P300, anhand derer das BCI sich für diesen Buchstaben entscheidet.
- **games** In dieser Anwendung werden die Spiele Snake, Pacman und Sokoban dargestellt. Die Bewegungen während des Spiels finden über Pfeile, die unterschiedlich aufleuchten, statt. Der Benutzer konzentriert sich auf einen Pfeil. Immer, wenn dieser Pfeil aufleuchtet, kommt es zu einer P300, anhand derer das BCI sich für diesen Pfeil, und die entsprechende Bewegung des Pfeils, entscheidet.
- **cursorControl** Mit dieser Anwendung wird ein Cursor dargestellt, der wie bei games auch über Pfeile die unterschiedlich aufleuchten, gesteuert werden kann.
- **BCIPractical** Im Fall der P300 entspricht diese Anwendung der matrixSpeller-Anwendung, mit dem Unterschied, dass hier statt einer Buchstabenmatrix eine Zahlenmatrix verwendet wird.

2. **IM**
   Die folgenden Anwendungen unterstützen das IM-Paradigma.

   - **imaginedMovement** In dieser Anwendung werden drei Punkte dargestellt und ein beweglicher Punkt in der Mitte dieser drei. Ziel bei dieser Anwendung ist es, den beweglichen Punkt auf einen von der Anwendung vorgegebenen Punkt zu bewegen. Mit folgenden Gedanken/Bewegungen wird dies realisiert:
     Gedanke: rechter Arm heben → Punkt bewegt sich nach rechts
     Gedanke: linker Arm heben → Punkt bewegt sich nach links
     Gedanke: beide Beine oder beide Arme heben → Punkt bewegt sich nach oben
   - **neurofeedback** Bei dieser Anwendung wird die ganze Zeit das Feedback, das die Anwendung bekommt, mit einer kontinuierlichen Bewegung eines Punktes dargestellt. Dieser Punkt wird wie der bewegliche Punkt beim imaginedMovement gesteuert.
   - **inducedDemo** Diese Anwendung entspricht prinzipiell der imaginedMovement-Anwendung. Mit dem Unterschied, dass es nur rechts und links gibt und der Punkt in der Mitte nicht beweglich ist. Es leuchten nur die jeweils gedachten Seiten auf.
   - **BCIPractical** Im Falle des IM entspricht diese Anwendung exakt der inducedDemo-Anwendung.

4.4.1.2 Anwendungen mit PTB

In dieser Aufzählung sind alle Anwendungen von Buffer-BCI, welche die PTB benötigen, aufgezählt.

1. **P300**
   Die folgende Anwendung unterstützt das P300-Paradigma.
   - **matrixSpellerPTB** Diese Anwendung ist genau wie die matrixSpeller-Anwendung, mit dem Unterschied, dass diese die PTB verwendet.
2. **SSVEP**
   Die folgende Anwendung unterstützt das SSVEP-Paradigma.
   - **ssep** Diese Anwendung stellt 4 Fenster dar, die jeweils mit unterschiedlichen Frequenzen aufleuchten. Diese sind gleichzeitig sichtbar und die Entscheidung, welches Feld der Benutzer angesehen hat, wird durch eine Frequenzanalyse des EEG getroffen. Diese Variante gilt als relativ robust.

4.4.2 Was beinhaltet Buffer-BCI → Die wichtisten Ordner kurz erklärt

In Abbildung 4.1 sind alle Ordner zu sehen, die Buffer-BCI zu Beginn mit sich bringt. Die Ordner BCIPractical, cursorControl, games, imaginedMovement, inducedDemo, matrixSpeller, matrixSpellerPTB, neurofeedback und ssep sind prinzipiell gleich aufgebaut. Jeder Ordner enthält die jeweils notwendigen MATLAB-Dateien für die Anwendung sowie Shell- bzw. Batchdateien zum Ausführen unter Mac bzw. Windows.

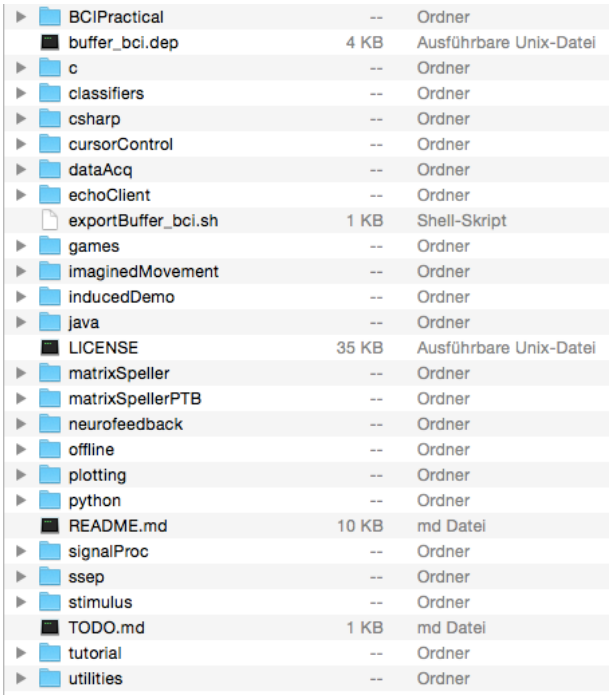


Abbildung 4.1: Das Framework Buffer-BCI dargestellt mit sämtlichen Dateien, die es enthält.

4.4.2.1 dataAcq

Dieser Ordner enthält Dateien, die für die Ausführung von Anwendungen mit Demo-Daten benötigt werden. Hierzu gehört unter anderem der SignalProxy, der simulierte Signale in den Buffer streamt. Des Weiteren enthält dieser Ordner Dateien, die für das Ansprechen der Hardware (Verstärker, A/D-Wandler verschiedener Fabrikate) nötig sind. Auch stellt dieser Ordner die notwendigen Dateien bereit, dass Buffer-BCI sprach- und platformunabhängig sein kann (unter anderem Dateien für Java, C, CSharp, Python, ...).

4.4.2.2 Anwendungen

Die Ordner der Anwendungen enthalten jeweils die Dateien, die für die Anwendung wichtig sind. Wie man diese Anwendungen startet wird in Abschnitt 4.6 beispielhaft an der matrixSpeller-Anwendung erläutert.

4.4.2.3 utilities

In diesem Ordner befinden sich unter anderem die Dateien, der Caps, die man auswählen kann. Die Caps die auswählbar sind, werden in Abbildung 4.2 darge-

stellt. Für die Experimente mit Buffer-BCI wurde während der Thesis immer die "cap_tmsi_mobita_p300.txt" verwendet.

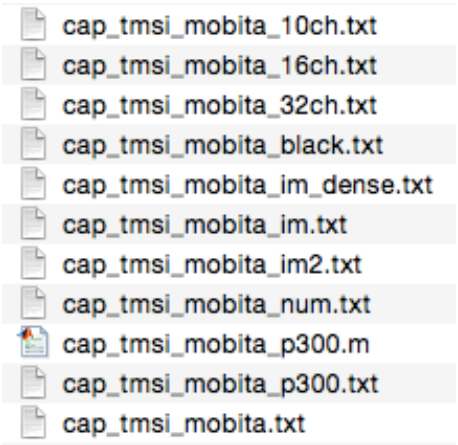


Abbildung 4.2: Diese Caps können für die verschiedenen Anwendungen ausgewählt werden

4.5 Konfiguration von tmsi_refa_config.txt

Bevor die matrixSpeller-Anwendung gestartet wird, muss diese Datei konfiguriert werden. Hierzu muss man das #-Zeichen vor den Elektroden die man auswählen möchte löschen (siehe Abschnitt 2.2.3 Abbildung 2.3).

4.6 Ablauf des P300 Experiments

In diesem Abschnitt wird am Beispiel der matrixSpeller-Anwendung erklärt, wie eine Anwendung von Buffer-BCI ausgeführt wird. Des Weiteren wird erklärt, wie genau diese Anwendung funktioniert und was der Benutzer tun muss.

4.6.1 Ausführen der Anwendung

Die Ausführung der Anwendung wird einerseits für die Verwendung mit Demo-Daten, andererseits für die Verwendung mit Echtzeitdaten erklärt. Mit einem Windows BS sollten jeweils die Batchdateien, mit einem Unix BS die Shelldateien ausgeführt werden.

- **mit Demo-Daten**
  1. Einen Buffer starten über dataAcq/startBuffer
  2. Datensimulation starten über dataAcq/startSignalProxy

3. Signalanalyseprozess starten über matrixSpeller/startSigProcBuffer
   Hinweis: warten, bis der Auswahldialog für die Cap erscheint.
4. Cap auswählen.
5. Anwendung starten über matrixSpeller/runSpeller

- **mit Echtzeitdaten**
  1. Den Echtzeitdatenbuffer starten über startTMSI Fieldtrip Buffer Refa
  2. Signalanalyseprozess starten über matrixSpeller/startSigProcBuffer
     Hinweis: warten, bis der Auswahldialog für die Cap erscheint.
  3. Cap auswählen.
  4. Anwendung starten über matrixSpeller/runSpeller

Nachdem runSpeller ausgeführt wurde, öffnet sich die Hauptgui (siehe Abbildung 4.3). Hier kann bei Subject ein Name eingegeben werden. Unter diesem Namen werden später die Trainingsdaten abgespeichert. **Hinweis:** Es muss darauf geachtet werden, dass der eingegebene Name keinen Punkt enthält, da alles, was nach dem Punkt geschrieben ist als Dateiendung verwendet wird und somit mit keinem Programm geöffnet werden kann.

Abbildung 4.3: Hauptgui der matrixSpeller-Anwendung

In der nachfolgenden Aufzählung werden die einzelnen Buttons der Hauptgui erläutert.

- **CapFitting**
  Bei CapFitting öffnet sich eine GUI, in der man die angeschlossenen Elektroden angezeigt bekommt, hier sollten möglichst alle Elektroden grün sein, was bedeutet, dass die Elektroden-Impedanz niedrig genug ist um eine gute Signalaufzeichnung zu bekommen.
- **EEG** Hier werden die aktuell aufgezeichneten Signale angezeigt mit dem Sinn, dass man überprüfen kann (z.B. anhand von Augenbewegungen oder Muskelanspannung (Zähne knirschen)) ob die Signale plausibel sind.
- **Practice** In Practice kann man, wenn man möchte, üben bevor man den Klassifikator kalibriert. Dieser Schritt ist optional.
- **Calibration** Hiermit wird eine Lernstichprobe erfasst, mit deren Hilfe der Klassifikator anschließend trainiert wird. Dem Benutzer wird die Buchstabenmatrix angezeigt. Er muss sich für die Erfassung der Lernstichprobe auf den angezeigten Buchstaben konzentrieren, bis die Erfassung beendet ist. Es ist hilfreich, wenn der Benutzer jedes mal, wenn der Buchstabe aufleuchtet, "ja" denkt.
- **Show Responses** Show Responses wurde selber hinzugefügt, um nach der Kalibrierung relevante Signalabschnitte, wie zum Beispiel die gemittelten Reize, anschauen zu können.
- **Train Classifier** Trainiert den Klassifikator für die Anwendungen.
- **Copy Spelling 1 & 2** Hier wird ein Wort vorgegeben, welches man nachbuchstabieren soll. Leider wurde noch nicht herausgefunden, wo dieses Wort vorgegeben werden kann.
- **Freespelling** Nachdem ein Klassifikator kalibriert und trainiert wurde, kann man beim Freespelling Wörter buchstabieren. Hierzu sucht sich der Benutzer einen Buchstaben heraus und konzentriert sich auf diesen. Auch hier ist es hilfreich, jedes mal, wenn der Buchstabe leuchtet, "ja" zu denken. Freespelling ist standardmäßig so eingestellt, dass nach 5 Buchstaben die Anwendung zu Ende ist.
- **ContFeedback** ist prinzipiell das Selbe, wie das Freespelling. Mit dem Unterschied, dass hier nicht nach 5 Buchstaben aufgehört wird.

In diesem Abschnitt, werden die System-Zustände sowie die System-Komponenten und deren Kommunikation beispielhaft von der matrixSpeller-Anwendung erläutert. Manche Anwendungen weichen etwas von der folgenden Beschreibung ab, das Prinzip ist jedoch stets das Selbe.

4.7 System-Zustände

Im Folgenden werden die Zustände erläutert, die das System annehmen kann. Jeder Zustand steht hierbei für einen Button aus der Hauptgui (siehe Abbildung 4.3). Anhand dieser Zustände werden von runSpeller aus jeweilige Events an den Buffer geschickt bzw. von startSigProcBuffer aus die einzelnen Funktionen aufgerufen.

Die Zustände können sein: capfitting, eegviewer, practice, calibrate/calibration, showresponses, train/classifier, copyspell1/2 und freespell.

Diese Zustände stehen dafür, was bei den Buttons passiert (siehe Abschnitt 4.6.1). Weswegen die einzelnen Zustände hier nicht weiter erklärt werden.

4.8 System-Komponenten

In der folgenden Auflistung werden die Komponenten dargestellt, die im Falle der matrixSpeller-Anwendung wichtig sind.

- **startSigProcBuffer** repräsentiert einen Client, der den Buffer auf bestimmte Ereignisse abhört und die entsprechenden EEG-Signalabschnitte aus dem Buffer anfordert und die entsprechenden Funktionen ausführt.
- **runSpeller** ist eine Funktion, die für die Versendung von Ereignissen an den Buffer zuständig ist. Des Weiteren kreiert diese Funktion die Hauptgui.
- **configureSpeller** ist die Konfigurationsdatei für die matrixSpeller-Anwendung. Hier kann unter anderem die Wiederholrate (siehe Abschnitt 5.2.4) eingestellt werden. Des Weiteren kann man hier festlegen, wie die Matrix aussehen soll.
- **controller** verwaltet die Ereignisse der Buttons aus der Hauptgui
- **spFeedbackStimulus** generiert die Zeilen- und Spaltenaufleuchtungen.
- **spCalibrateStimulus** erstellt die Sequenz, die unter Calibration verwendet wird.

4.9 Kommunikation der Komponenten

Die Grundlage für die Kommunikation zwischen den Komponenten bildet der FieldTrip-Buffer (Abschnitt 2.2.1). Die Kommunikation zwischen Buffer und Funktionen basiert auf TCP/IP-Sockets. Funktionen senden anhand der Zustände des Programms Ereignisse an den Buffer. Dieser reagiert auf diese Ereignisse und spricht weitere Funktionen an. Die Bufferkommunikation wird anhand des Beispiels “Zeile/Spalte aufleuchten und jeweiliges Ereignis senden“ in Abbildung 4.4 dargestellt.

Die Kommunikation wird an folgender Grafik verdeutlicht bzw. vereinfacht dargestellt.

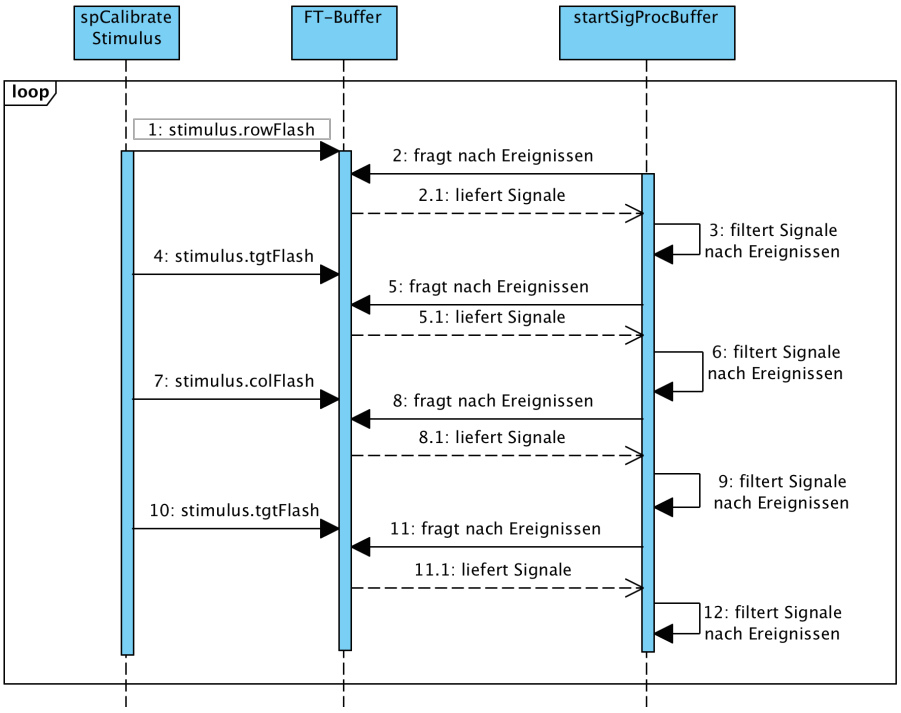


Abbildung 4.4: Kommunikation der Komponenten, dargestellt am Beispiel "Zeile/Spalte aufleuchten und jeweiliges Ereignis senden"

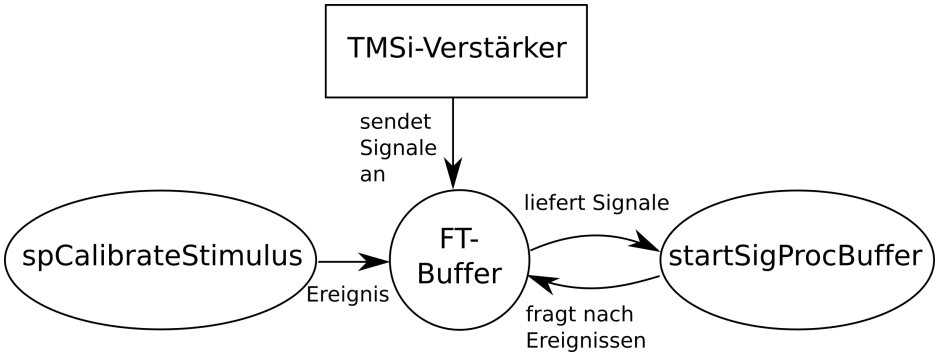


Abbildung 4.5: Verdeutlichung bzw. vereinfachte Darstellung der Bufferkommunikation

5 Eigene praktische Experimente mit Buffer-BCI

In diesem Kapitel wird vorgestellt, welche eigenen Experimente mit Buffer-BCI durchgeführt wurden. Des Weiteren werden die Latenzmessung sowie die Probandenexperimente näher erläutert. Diese Experimente wurden durchgeführt, um abschätzen zu können, ob Buffer-BCI für eine Praktikumsaufgabe in Frage kommt.

5.1 Abschätzung der Latenzzeiten bei der Signalaufzeichnung

Zwischen der Signalaufzeichnung (TMSi-Verstärker → FieldTrip-Buffer) einerseits und der Signalisierung eines Ereignisses (Buffer-BCI-Komponenten → FieldTrip-Buffer) andererseits kommt es zu einem Versatz. Dieser darf nicht zu groß sein, damit z.B. eine vom Verstärker erfasste P300-Welle im tatsächlich erwarteten Zeitfenster (bis 300 ms nach Trigger-Ereignis) zum liegen kommt. Andernfalls würde man in den aufgezeichneten Signalen an der falschen (zeitlichen) Stelle suchen.

5.1.1 Messaufbau

In Abbildung 5.1 ist dargestellt, mit welchem Messaufbau die Latenzmessung durchgeführt wurde. Mit Hilfe des Digital I/O wurden die Bits gesetzt. Dieser ist mit dem Oszilloskop verbunden, um die Dauer der Bitänderung messen zu können. Der TMSi-Verstärker ist in den Messaufbau integriert, um eine Darstellung von Signalaufzeichnung und Ereignissignalisierung erstellen zu können.

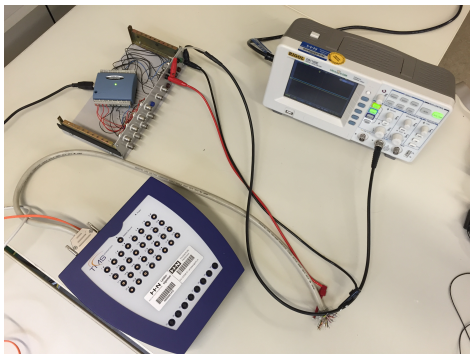
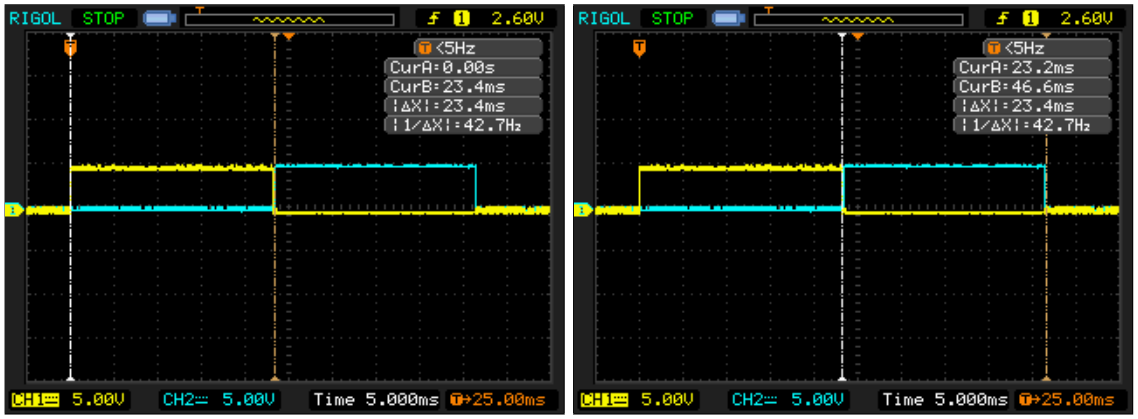


Abbildung 5.1: Messaufbau für Latenzmessung: oben links ist der Digital I/O, rechts daneben das Oszilloskop

5.1.2 Realisierung der Messung

Nacheinander werden zwei Bits gesetzt um den Zeitbedarf für die Bitänderung abzuschätzen(dies wurde mit dem Oszilloskop gemessen siehe Abbildung 5.2. Danach wurde mit diesen Bits signalisiert, wann in der matrixSpeller-Anwendung eine Zeile/Spalte zum Leuchten gebracht wurde und wann das entsprechende Ereignis in den Buffer geschrieben wurde. Hierzu wird das entsprechende Bit mit dem Trigger-Eingang des Verstärkers verbunden (der wiederum synchron mit den EEG-Daten aufgezeichnet wird). Anschließend wird der Trigger-Kanal zusammen mit den aufgezeichneten Ereignissen dargestellt. Anhand dieser Darstellung (Abbildung 5.3) lässt sich der Zeitversatz abschätzen.



(a) Bitänderung 1.Bit (b) Bitänderung 2.Bit

Abbildung 5.2: Dauer der Bitänderung

An diesen Grafiken (Abbildung 5.2) ist zu sehen, dass die Bitänderung für jedes Bit ca. 25 Millisekunden beträgt.

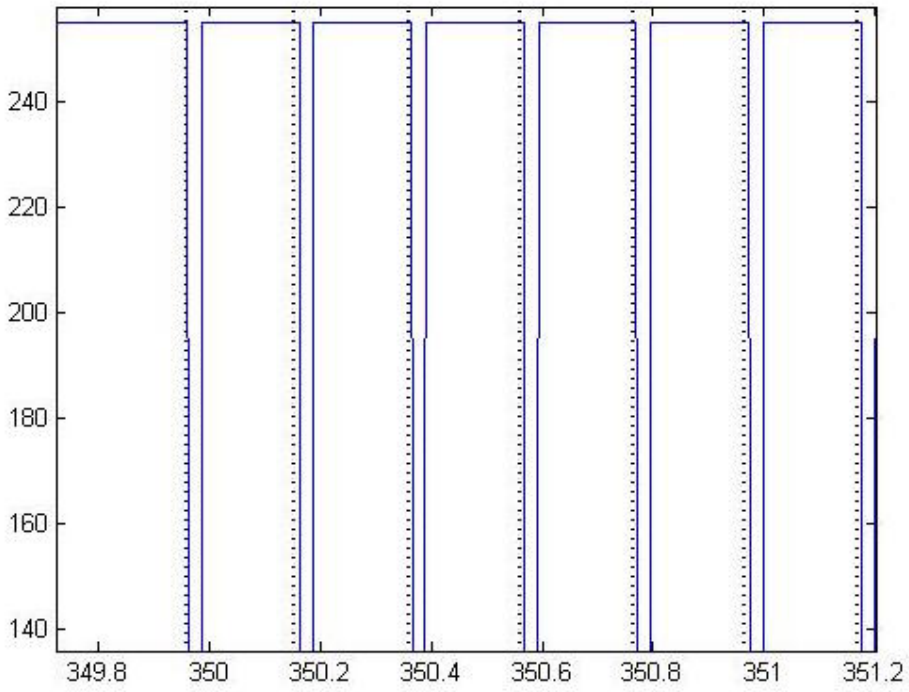


Abbildung 5.3: Versatz zwischen Signalaufzeichnung und Ereignissignalisierung

Anhand dieser Grafik (Abbildung 5.3) zeigt sich, dass beim ersten Event(bei ca. 349.9) die Signalaufzeichnung und Ereignissignalisierung zeitgleich stattfinden. Bei den weiteren Events kommt es jeweils zu einem kleinen Versatz zwischen Signalaufzeichnung und Ereignissignalisierung. Wenn man bedenkt, dass die Bitänderung ca. 25 Millisekunden beträgt, lässt sich hier abschätzen, dass es zu einem Versatz von ca. 23,4 Millisekunden kommt. Dieser Versatz ist jedoch so minimal, dass er im Falle einer P300-Anwendung vernachlässigt werden kann.

5.1.3 Fazit der Messung

Die Messung wurde an 2 unterschiedlichen Computern durchgeführt. Hierbei zeigte sich, dass die Latenz durchaus Computerabhängig sein kann. Bei dem einen Computer (intel i7, 64-Bit Windows 8.1, 16 GB Hauptspeicher) beträgt die Latenz ca. 25 Millisekunden (Messung die zuvor dargestellt wurde). Bei dem anderen Computer (intel Core2Duo 2 Ghz, 32-Bit Windows 7, 2 GB Hauptspeicher) ist die Latenz fast doppelt so groß und beträgt ca. 45 Millisekunden.

Die Messung zeigte, dass es zu einem minimalen Versatz zwischen Signalaufzeichnung und Ereignissignalisierung kommt. Dieser kann jedoch vernachlässigt werden.

5.2 P300-Speller

5.2.1 Elektrodenkonfiguration

Die Elektroden wurden folgendermaßen angeschlossen:

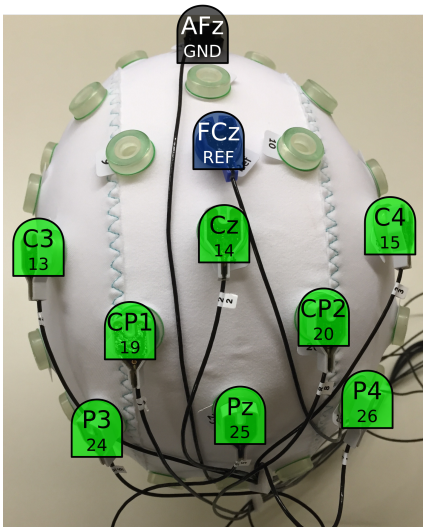


Abbildung 5.4: Zuweisung der Elektroden auf der actiCap

Elektrode	Halterung
GND	AFz (GND)
REF	FCz (REF)
1	C3 (13)
2	Cz (14)
3	C4 (15)
4	P4 (26)
5	Pz (25)
6	P3 (24)
7	CP1 (19)
8	CP2 (20)

Tabelle 5.1: Zuweisung der Elektroden an Halterungen

Die Anordnung der Elektroden wurde so gewählt, da die P300-Welle bei den parietalen Elektroden (Pz) ihr Maximum hat. Es werden 8 Elektroden verwendet, da dies zum einen ausreichend ist und zum anderen eine große Zeitersparnis beim Anschließen einer Person an das System mit sich bringt. Insbesondere das Befüllen der Elektroden mit Elektrolyt Gel kann als ungeübte Person, wie man sie im PSB erwarten kann, einige Zeit in Anspruch nehmen.

5.2.2 Zeitaufwandsmessungen

Um den Zeitaufwand der Praktikumsaufgabe grob abschätzen zu können, wurde bei unterschiedlichen Probanden die Zeit gemessen, die benötigt wird um 8 + 2(GND & REF) Elektroden zu befüllen. Das Ergebnis dieser Messung ist in der folgenden Tabelle zu sehen.

Proband	Zeit
1	8:40
2	5:03
3	8:22

Tabelle 5.2: Dauer zum Befüllen der Elektroden

Durchschnittlich braucht man ca. 7:22 Minuten für das Befüllen von 10 Elektroden.

5.2.3 Konfiguration von tmsi_refa_config.txt

Diese Datei muss entsprechend den ausgewählten Elektroden konfiguriert werden. Hierzu muss man das #-Zeichen vor den Elektroden die man auswählen möchte löschen (siehe Abschnitt 2.2.3 Abbildung 2.3).

5.2.4 Variation der Anzahl Wiederholungen

Es wurde untersucht, ob man an der Anwendung selber irgendeinen Einfluss auf den Verlauf des Experiments nehmen kann. Dabei kam heraus, dass man die Zeit die es braucht um einen Buchstaben zu kalibrieren erheblich verkürzen kann. In Tabelle 5.3 ist dargestellt, wie lange es dauert einen Buchstaben zu kalibrieren. Wenn man überlegt die längste Zeit pro Buchstabe zu benötigen, und ein Wort mit beispielsweise 5 Buchstaben buchstabieren zu wollen, kann das ganz schön anstrengend werden, wenn man sich so lange auf einen Buchstaben konzentrieren muss.

Wiederholrate	Zeit (Minuten)
5	1:11
4	0:59
3	0:47
2	0:34
1	0:21
0.5	0:15
0.25	0:12

Tabelle 5.3: zeitliche Dauer mit unterschiedlichen Wiederholraten pro Buchstabe

Um die Zeit für die Kalibrierung eines Buchstaben verkürzen zu können, wird die Wiederholrate der Zeilen- und Spaltenaufleuchtungen minimiert. Dies kann in configureSpeller mit der Variable nRepetitions eingestellt werden. Anhand von Probanden wurde getestet, ob auch bei einer minimierten Wiederholrate noch richtige Ergebnisse herauskommen(siehe Abschnitt 5.2.5).

5.2.5 Reproduzierbarkeit

Um zu testen, ob die matrixSpeller-Anwendung auch bei vielen verschiedenen Personen vergleichbare Ergebnisse liefert, wurden Probandentests durchgeführt. Die Aufgabe war bei unterschiedlichen Wiederholraten (5.2.4) jeweils das Wort "HALLO" zu buchstabieren. Die Ergebnisse der Probandentests sind in folgender Tabelle dargestellt:

Proband	Wiederholrate	H	A	L	L	O	Ergebnis
1	5	✓	✓	✓	✓	✓	5/5
	4	✓	✓	✓	✓	✓	5/5
	3	✓	✓	✓	✓	✓	5/5
	2	✓	✓	✓	✓	✓	5/5
	1	✓	✓	✓	✓	✓	5/5
	0.5	✗	✓	✓	✓	✓	4/5
	0.25	✗	✓	✓	✗	✓	3/5
2	5	✓	✓	✓	✓	✓	5/5
	4	✓	✓	✓	✓	✓	5/5
	3	✓	✓	✓	✓	✓	5/5
	2	✓	✓	✓	✓	✓	5/5
	1	✓	✓	✓	✓	✓	5/5
	0.5	✗	✓	✗	✓	✓	3/5
	0.25	✗	✓	✓	✓	✗	3/5
3	5	✓	✓	✓	✓	✓	5/5
	4	✓	✓	✓	✓	✓	5/5
	3	✓	✓	✓	✓	✓	5/5
	2	✓	✓	✓	✓	✓	5/5
	1	✓	✓	✓	✓	✓	5/5
	0.5	✗	✓	✓	✓	✗	3/5
	0.25	✗	✓	✗	✗	✓	2/5

Tabelle 5.4: matrixSpeller-Ergebnis bei unterschiedlichen Wiederholraten

Wiederholrate	gemitteltes Ergebnis
5	5/5
4	5/5
3	5/5
2	5/5
1	5/5
0.5	3/5
0.25	2/5

Tabelle 5.5: gemitteltes Ergebnis für verschiedene Wiederholraten

Es lässt sich sagen, dass die Reproduzierbarkeit auf jeden Fall gegeben ist. Anhand von Tabelle 5.5 sieht man, dass eine Wiederholrate von 0,5 oder 0,25 besser gemieden werden sollte, da hier deutlich mehr Fehler auftreten.

5.3 Steady State Visual Evoked Potential

Beim SSVEP gab es zuerst einmal das Problem, dass die aufgezeichneten Daten nicht gespeichert wurden. Dieses Problem wurde mittlerweile gelöst, durch hinzufügen eines Zustands (calibrationptb) in startSigProcBuffer. Bei der ssep-Anwendung war nicht zu erkennen, wie sie funktionieren hätte sollen. Nach einiger Überlegung wurde beschlossen, nicht darauf zu schauen, was die Anwendung tun soll, sondern zu überprüfen, ob die Frequenzen mit denen die 4 Felder flackern, im EEG zu sehen sind. Nachdem dies nach mehreren Versuchen nie der Fall war, wurde beschlossen diese Anwendung bei Seite zu legen.

5.4 Imagined Movement

Die IM-Anwendungen haben in soweit funktioniert, dass sie nicht abgestürzt sind und die Daten gespeichert wurden. Hier war es eher das Problem, dass nicht richtig nachvollzogen werden konnte, ob das Programm das gemacht hat, was man sich vorstellte, oder ob es Zufall war, als die richtigen Punkte angesteuert wurden. Somit wurden diese Anwendungen im weiteren Verlauf der Thesis nicht mehr verfolgt.

5.5 Fallstricke

Bei der Durchführung der Experimente ist es öfter zu Problemen gekommen. Diese werden in der folgenden Aufzählung erläutert:

1. **findMatlab** Zu Beginn wurde die Anwendung mit dem "falschen" MATLAB gestartet. Um dies zu verhindern wurde die findMatlab-Datei folgendermaßen angepasst: In Abbildung 5.5 ist der relevante Ausschnitt aus der Originaldatei zu sehen. In Abbildung 5.6 der abgeänderte Ausschnitt der Datei.

```
in ("C:\Program Files\MATLAB\*" "C:\Program Files (x86)\MATLAB\*")
```

Abbildung 5.5: findMatlab Original

```
in ("C:\Program Files\MATLAB\*")
```

Abbildung 5.6: findMatlab geändert

2. **actiCap-Adapter** Es scheint so, als hätte der actiCap-Adapter einen Wackelkontakt. Nach dem Befüllen der Elektroden wiesen alle eine niedrige Impedanz auf (leuchteten grün). Im CapFitting jedoch wurden alle Elektroden als rot (zu hohe Impedanz) angezeigt. Nach wackeln am actiCap-Adapter bzw. dessen Kabel am Verstärker wurden die Elektroden beim CapFitting auch grün.

3. **Elektroden befüllen** Beim Befüllen der Elektroden sollte darauf geachtet werden, dass mit der GND und REF begonnen wird. Erst wenn diese beiden eine niedrige Impedanz aufweisen (grün leuchten), sollten die Datenelektroden befüllt werden.

4. **Impedanzmodus an der Controlbox** Wenn man Signale aufnehmen möchte, muss man darauf achten, dass der Impedanzmodus an der Controlbox, nach dem Befüllen der Elektroden, deaktiviert wird. Da die Messung sonst keine plausiblen Signale liefert.

5. **subjectName** Wie bereits in 4.6.1 erwähnt muss hier darauf geachtet werden, dass der Name des Subjekts keinen Punkt enthält. Alles was hinter dem Punkt im Subjektname steht wird als Dateiendung der Trainingsdatendatei verwendet und kann nicht mehr geöffnet werden.

6 Praktikumsversuch

6.1 Was soll vermittelt werden? → Lernziele

Anhand eines Praktikumsversuchs soll den Studierenden das Prinzip von BCIs näher gebracht werden. Sie sollen verstehen, wie ein BCI funktioniert und welche Grenzen es bei diesen gibt. Sie sollen die Möglichkeiten kennen lernen, wie man Programme nur mit Gedanken steuern kann und mit welcher Technik (Hardware) dies realisiert wird. Für diesen Versuch soll das Prinzip eines BCIs mittels einer P300-Reizantwort verstanden werden.

6.2 Erweiterung um Anzeige der Reizantworten

Das Problem war, dass Buffer-BCI keine Möglichkeit bot, die Reize, wie z.B. die P300-Welle, welche für die Entscheidungen benutzt wird, zu visualisieren. Darum wurde das Framework um eine Anzeige der Reizantworten erweitert. Mit Hilfe dieser Anzeige ist es möglich, den Studierenden die relevanten Signalsegmente, wie zum Beispiel die P300, im Praktikumsbetrieb veranschaulichen zu können. Es handelt sich dabei um eine Anzeige der jeweils gemittelten Reizantworten von Zielreiz und Standardreiz. Diese werden in verschiedenen GUIs präsentiert.

6.2.1 Graphical User Interfaces

Es wurden für 3, 8, 16 oder 32 angeschlossene Elektroden jeweilige GUIs erstellt.

- **3 angeschlossene Elektroden** Diese Graphical User Interface (GUI) wird benötigt, wenn man die Anwendung mit Demo-Daten startet, da in diesem Fall 3 Elektroden simuliert werden.

Abbildung 6.1: GUI für 3 Elektroden

- **8 angeschlossene Elektroden** Diese GUI wird benötigt, wenn man die Anwendung mit 8 angeschlossenen Elektroden startet. Diese wurde für die eigenen praktischen Experimente (5) verwendet, da diese mit 8 Elektroden durchgeführt wurden. Die Anzahl von 8 Elektroden ist auch gut für das PSB geeignet, da diese GUI im Gegensatz zu der mit 32 Elektroden (Abbildung 6.3) noch übersichtlich ist.

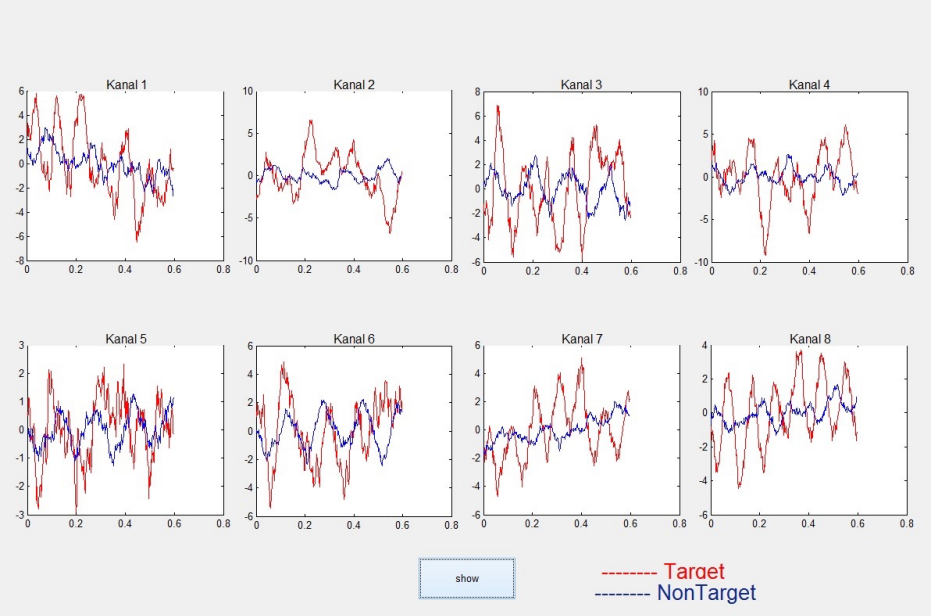


Abbildung 6.2: GUI für 8 angeschlossene Elektroden

- **16 oder 32 angeschlossene Elektroden** Wenn ein Benutzer mehr Elektroden verwenden möchte, ist ihm die Möglichkeit der Anzeige der Reizantworten noch für 16 und 32 Elektroden gegeben. Wie allerdings in Abbildung 6.3 zu sehen, können so viele Elektroden in der Anzeige schnell unübersichtlich werden. Auf die Darstellung der GUI von 16 Elektroden wird hier verzichtet, da diese der linken Hälfte von Abbildung 6.3 entspricht.

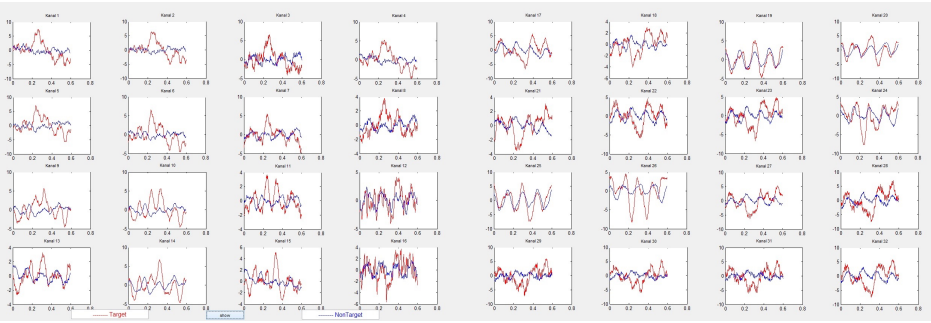


Abbildung 6.3: GUI für 32 angeschlossene Elektroden

6.2.2 Programmcode

In der folgenden Aufzählung wird erläutert, was in welcher Funktion ergänzt, beziehungsweise welche Funktionen neu geschrieben werden mussten.

6.2.2.1 selbstgeschriebene Funktionen

Die folgenden Funktionen wurden selbst geschrieben und zum Buffer-BCI-Framework hinzugefügt.

- **matrixSpeller/splitTargetNonTarget** Diese Funktion nimmt die Events (Zielreiz = target, Standardreiz = nonTarget) aus den traindevents des Buffers. Mit Hilfe dieser Funktion werden die Zielreize und die Standardreize anhand von value (siehe Zeile 34 und 48 in Listing 6.1) für die spätere Anzeige der Reizantworten separiert. Wenn die value 1 beträgt, handelt es sich um einen Zielreiz. Bei einer value von 0 handelt es sich um einen Standardreiz.

```
1  function [NonTarget , Target] = splitTargetNonTarget(traindevents)
2
3  fieldNT = 'number';
4  fieldT = 'number';
5  fieldNT1 = 'value';
6  fieldT1 = 'value';
7  fieldNT2 = 'sample';
8  fieldT2 = 'sample';
9  fieldNT3 = 'offset';
10 fieldT3 = 'offset';
11 fieldNT4 = 'duration';
12 fieldT4 = 'duration';
13
14 valueNT = {'something'; 'ab'; 'ab'};
15 valueT = {'something'; 'ab'; 'ab'};
16 valueNT1 = {'something'; 'ab'; 'ab'};
17 valueT1 = {'something'; 'ab'; 'ab'};
18 valueNT2 = {'something'; 'ab'; 'ab'};
19 valueT2 = {'something'; 'ab'; 'ab'};
20 valueNT3 = {'something'; 'ab'; 'ab'};
21 valueT3 = {'something'; 'ab'; 'ab'};
22 valueNT4 = {'something'; 'ab'; 'ab'};
23 valueT4 = {'something'; 'ab'; 'ab'};
```

```
24
25 NonTarget = struct(fieldNT,valueNT,fieldNT1,valueNT1,fieldNT2,valueNT2,
26                     fieldNT3,valueNT3,fieldNT4,valueNT4);
27 Target = struct(fieldT,valueT,fieldT1,valueT1,fieldT2,valueT2,fieldT3,
28                     valueT3,fieldT4,valueT4);
29
30 counterTarget = 1;
31 counterNonTarget =1;
32
33 for i =1:length(traindevents);
34     if traindevents(i).value == 1;
35         Target(counterTarget).number = i;
36         Target(counterTarget).value = traindevents(i).value;
37         Target(counterTarget).sample = traindevents(i).sample;
38         Target(counterTarget).offset = traindevents(i).offset;
39         Target(counterTarget).duration = traindevents(i).duration;
40         counterTarget = counterTarget+1;
41     elseif traindevents(i).value == 0;
42         NonTarget(counterNonTarget).number = i;
43         NonTarget(counterNonTarget).value = traindevents(i).value;
44         NonTarget(counterNonTarget).sample = traindevents(i).sample;
45         NonTarget(counterNonTarget).offset = traindevents(i).offset;
46         NonTarget(counterNonTarget).duration = traindevents(i).duration;
47         counterNonTarget = counterNonTarget+1;
48     end
49 end
```

Listing 6.1: splitTargetNonTarget

- **matrixSpeller/meanSignal** Um die Reizantworten zu mitteln wird diese Funktion verwendet.

```
1  function [mMeanTarget, caSingleSweeps] = meanTarget(traindata, Target)
2
3  ti = 1;
4  mMeanTarget = zeros(size(traindata(ti).buf));
5
6  nNumChan = size(traindata(ti).buf, 1);
7  for ci = 1:nNumChan
8      mTargetCi = [];
9      for j = 1:length(Target)
10         vCurrentSweep = traindata(Target(j).number).buf(ci,:);
11         mTargetCi(j,:) =  vCurrentSweep - mean(vCurrentSweep);
12     end
13     caSingleSweeps{ci} = mTargetCi;
14     mMeanTarget(ci,:) = mean(mTargetCi);
15 end
```

Listing 6.2: meanSignal

- **utilities/showResponses** Für die Anzeige der jeweiligen GUIs ist diese Funktion verantwortlich. Sie lädt die Daten, die auf dem Computer gespeichert wurden. Mit Hilfe eines Index i wird die Anzahl der Elektroden ermittelt. Anhand dieses Index wird mit Hilfe eines Switch-Case-Statements die Entscheidung getroffen, welche GUI angezeigt werden soll. Ein Statement wird Beispielhaft in Listing 6.3 dargestellt.

```
function [varargout]=showResponses(varargin)
load C:\userdata\shared\buffer_bci-master\matrixSpeller\test.mat
load current
i = size(traindata(1).buf, 1);

switch i
    case 3
        contFig=myFig3();
            while (ishandle(contFig))
                set(contFig,'visible','on');
                uiwait(contFig); % CPU hog on ver 7.4
                    if ( ~ishandle(contFig) ) break; end;
                set(contFig,'visible','off');
            end
```

Listing 6.3: Beispielhafte Darstellung eines Switch-Case-Statements: In diesem Fall für 3 angeschlossene Elektroden

- **figures** Für jede GUI wurde eine eigene Figure in Matlab kreiert. In Listing 6.4 ist der wichtigste Aufruf der Figurefunktion für 3 Elektroden dargestellt.

```
% ---- Executes on button press in show.
function show_Callback(hObject, eventdata, handles)
% hObject    handle to show (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load /Users/bea/Desktop/buffer_bci-master/matrixSpeller/test.mat
load current
[NonTarget, Target] = splitTargetNonTarget(traindevents);
vTime = [0:length(traindata(1).buf)-1]/512;
[mMeanTarget] = meanSignal(traindata,Target);
axes(handles.axes1);
plot(vTime, mMeanTarget(1,:),'r')
hold
[mMeanNonTarget] = meanSignal(traindata,NonTarget);
plot(vTime, mMeanNonTarget(1,:),'b')
axes(handles.axes2);
plot(vTime, mMeanTarget(2,:),'r')
hold
[mMeanNonTarget] = meanSignal(traindata,NonTarget);
plot(vTime, mMeanNonTarget(2,:),'b')
axes(handles.axes3);
plot(vTime, mMeanTarget(3,:),'r')
hold
[mMeanNonTarget] = meanSignal(traindata,NonTarget);
plot(vTime, mMeanNonTarget(3,:),'b')
```

Listing 6.4: Beispielhafte Darstellung einer Figurefunktion: In diesem Fall für 3 angeschlossene Elektroden

6.2.2.2 ergänzte Funktionen

Die folgenden Funktionen wurden für die Anzeige der Reizantworten um den angezeigten Code in den jeweiligen Listings ergänzt.

- **controller** Im Controller musste eingefügt werden, was bei Klick des Buttons showResponses passieren soll, wie in Listing 6.5 zu sehen.

```
1 % --- Executes on button press in showresponses.
2 function showresponses_Callback(hObject, eventdata, handles)
3 % hObject    handle to showresponses (see GCBO)
4 % eventdata  reserved - to be defined in a future version of MATLAB
5 % handles    structure with handles and user data (see GUIDATA)
6 handles.phaseToRun=get(hObject,'Tag');
7 guidata(hObject,handles);
8 uiresume;
```

Listing 6.5: controller

- **runSpeller** In runSpeller wurde showResponses als neuer Zustand hinzugefügt (Listing 6.6). Es werden mit dieser Funktion entsprechende Events an den Buffer gesendet.

```
1     case 'showresponses';
2     sendEvent('subject',info.subject);
3     sendEvent('startPhase.cmd',phaseToRun);
4     % wait until capFitting is done
5     buffer_newevents(buffhost,buffport,[],phaseToRun,'end');
```

Listing 6.6: runSpeller

- **startSigProcBuffer** Dieser Zustand musste zusätzlich auch in startSigProcBuffer hinzugefügt werden (Listing 6.7).

```
1 case 'showresponses';
2 showResponses(buffhost,buffport,'capFile',capFile,'overridechnms',
3 overridechnms);
```

Listing 6.7: startSigProcBuffer

6.3 Versuchsanleitung

Für das PSB wurde eine Versuchsanleitung mit einem P300-Speller-Experiment erstellt. Diese befindet sich im Anhang. Sie kann noch um SSVEP- und IM-Experimente erweitert werden.

7 Zusammenfassung und abschließende Bewertung

Für das Praktikum Medizinische Signal- und Bildverarbeitung (PSB) des Studiengangs Medizinische Informatik sollte eine Praktikumsaufgabe mit BCI-Konzepten realisiert und entwickelt werden. Hierfür sollte die im Labor neu verfügbare Hardware (actiCap-System, TMSi-Verstärker(siehe Abschnitt 2.3)) und das Framework Buffer-BCI verwendet werden. Bevor jedoch ein Experiment für das PSB erstellt werden konnte, mussten einige Randbedingungen geklärt werden. Hierzu zählte unter anderem Buffer-BCI auf seine Eignung für eine Praktikumsaufgabe zu untersuchen. Wichtige Teilergebnisse waren hier die Anforderungsanalyse(Kapitel 7), die Untersuchung von Buffer-BCI, die Durchführung eigener praktischer Elemente und die Erstellung des Praktikumsversuchs. Rückblickend lässt sich hier sagen, dass diese Teilergebnisse wichtig waren, da Buffer-BCI ein komplexes Framework mit spärlicher Dokumentation ist. Daher war auch die lange Einarbeitungszeit in Buffer-BCI nötig. Eine große Hilfe war es, dass Buffer-BCI mit Demo-Daten verwendet werden konnte, wodurch es nicht zwingend notwendig war immer die Hardware aufbauen und verwenden zu müssen. Und es so auch möglich war, von zu Hause aus Funktionen für Buffer-BCI schreiben und testen zu können.

7.1 Buffer-BCI

Buffer-BCI war definitiv die richtige Wahl um Experimente für das Praktikum erstellen zu können. Bei ersten Tests zeigte sich bereits, dass es deutlich besser funktionierte als andere Frameworks. Buffer-BCI bietet Anwendungen zu den verschiedenen BCI-Paradigmen P300, IM und SSVEP. Leider konnte nur mit den Anwendungen, die das P300-Paradigma verwenden, gut gearbeitet werden. Bei den IM-Anwendungen konnte nicht nachvollzogen werden, ob die Experimente richtig durchgeführt wurden, weshalb diese nicht weiter verfolgt wurden. Das Gleiche galt auch für die SSVEP-Anwendung. Auch diese wurde nicht weiter verfolgt. Zum Einen, wegen der Funktionalität, die hier nicht immer gegeben war, zum Anderen weil auch hier nicht nachvollzogen werden konnte, ob das Experiment richtig durchgeführt wurde.
Die P300-Anwendungen haben nach ein paar Änderungen weitestgehend funktioniert und konnten daher für das Praktikum verwendet werden. Es muss gesagt werden, dass sich Buffer-BCI noch in der Entwicklung befindet. Es kommen neue Funktionen hinzu und es werden Verbesserungen von bisherigen Funktionen vorgenommen. Auch die Dokumentation wird erweitert und verbessert. Hat man eine lauffähige Version, sollte darauf geachtet werden diese zu sichern, da nach einem Update die Funktionstüchtigkeit nicht mehr gewährleistet ist. Aufgrund der spärlichen Dokumentation von Buffer-BCI war eine sehr lange Einarbeitungszeit notwendig. Diese wurde vor allem durch die Tatsache verursacht, dass das Framework viele unterschiedliche Programmiersprachen verwendet um eine Platformunabhängigkeit gewährleisten zu können.

7.2 Hardware

Die Hardware, die im Labor verfügbar ist, konnte nach minimalen softwaretechnischen Änderungen einwandfrei mit dem Framework Buffer-BCI interagieren. Die Einarbeitungszeit für die Hardware war aufgrund guter Dokumentation deutlich geringer als die Einarbeitungszeit in Buffer-BCI. Die Hardware eignete sich gut für die durchgeführten Experimente. Durch die actiCap wird das Anbringen der Elektroden auf der Kopfhaut erheblich vereinfacht. Auch die Kontrolle, ob die Elektroden-Impedanz niedrig genug ist um einwandfreie Signale aufzeichnen zu können, ist durch eine Impedanzmessung der Elektroden einfach und intuitiv gestaltet. Lediglich beim actiCap-Adapter tauchte das Problem auf, dass dieser einen Wackelkontakt bei der Verbindung zum TMSi-Verstärker hat.

7.3 Praktische Experimente

Um das System besser kennen zu lernen waren eigene praktische Experimente hilfreich. Es wurden alle BCI-Konzepte untersucht. Des Weiteren wurden Latenzen gemessen, um die zeitliche Verzögerung zwischen Signalaufzeichnung und Ereignissignalisierung abschätzen zu können. Um die Reproduzierbarkeit der Ergebnisse der Anwendungen belegen zu können, wurden Probandentests durchgeführt.

7.3.1 durchgeführte Experimente

Bei den durchgeführten Experimenten stellte sich heraus, dass SSVEP- und IM-Experimente nicht für eine Aufgabe im Praktikum geeignet sind. Da es bei diesen beiden Konzepten öfter zu Programmabstürzen kommt und man auch nicht nachvollziehen konnte, ob die Anwendung funktioniert hat, wurden diese beiden Experimente nicht weiter verfolgt.

Im Gegensatz dazu funktionierte die matrixSpeller-Anwendung einwandfrei. Das bedeutet, Wörter konnten buchstabiert werden, die Trainingsdaten wurden gespeichert und es kam zu keinen Programmabstürzen. Bei den Experimenten wurde festgestellt, dass eine Veranschaulichung der relevanten Signalabschnitte (Zielreiz, Standardreiz) fehlte. Daher wurde beschlossen diese selber zu entwickeln (siehe Abschnitt ...).

7.3.2 Latenzmessung

Zwischen der Signalaufzeichnung (TMSi-Verstärker → FieldTrip-Buffer) einerseits und der Signalisierung eines Ereignisses (Buffer-BCI-Komponente → FieldTrip-Buffer) andererseits kommt es zu einem Versatz. Dieser darf nicht zu groß sein, damit z.B. eine vom Verstärker erfasste P300-Welle im tatsächlich erwarteten Zeitfenster (bis 300 ms nach Trigger-Ereignis) zum liegen kommt. Andernfalls würde man in den aufgezeichneten Signalen an der falschen (zeitlichen) Stelle suchen.

Um diese Messung realisieren zu können, war ein geeigneter Messaufbau notwendig. Dafür wurden mit einem Digital I/O nacheinander zwei Bits gesetzt, um den Zeitbedarf für die Bitänderung abschätzen zu können. Dieser Zeitbedarf wurde mit einem Oszilloskop gemessen. Um die gesetzten Bits zusammen mit den Signalen darstellen zu können, musste auch der TMSi-Verstärker mit angeschlossen werden.

Bei dieser Messung kam heraus, dass es zu einem Versatz zwischen Signalaufzeichnung und Ereignissignalisierung von ca. 23,4 Millisekunden kommt. Dieser Versatz ist so klein, dass er im Falle der P300 vernachlässigt werden kann.

7.3.3 Probandentests

Um die Reproduzierbarkeit des Systems untersuchen zu können, wurden Probandentests durchgeführt. Die matrixSpeller-Anwendung wurde mit 3 Probanden bei unterschiedlichen Wiederholraten (programmspezifischer Parameter) getestet. Die Wiederholrate gibt an, wie viele Zeilen/Spalten aufleuchten. Es wurde getestet, ab welcher Wiederholrate es zu Fehlern kommt, indem die Probanden ein Wort buchstabieren sollten. Generell funktioniert die matrixSpeller-Anwendung für größere Wiederholraten durchgängig fehlerfrei. Jedoch führen längere Wiederholraten zur Ermüdung. Kürzere Zeiten sind bis zu einer Wiederholrate von 1 stabil. Bei einer Wiederholrate kleiner 1 steigt die Fehleranfälligkeit. Für einen Praktikumsversuch ergibt sich daraus, dass unterschiedliche Wiederholraten ausprobiert werden sollten, um ein Gefühl dafür zu bekommen, ab welcher Wiederholrate ein sinnvolles Ergebnis geliefert wird.

7.4 Erweiterung von Buffer-BCI

Da das Framework Buffer-BCI keine Möglichkeit geboten hat, sich die relevanten Signalabschnitte wie z.B. die P300-Welle anschauen zu können, wurde eine GUI erstellt, die diese Möglichkeit bietet. Mit Hilfe der GUI können für jede Elektrode die gemittelte Reizantwort von Ziel- bzw. Standardreiz angezeigt werden. Dies ist wichtig, damit die Studierenden zum einen nachvollziehen können, wie ein BCI funktioniert und zum anderen den Unterschied zwischen Ziel- und Standardreiz sehen können. Für die gängigsten Anzahlen (3, 8, 16 und 32) von angeschlossenen Elektroden wurde jeweils eine GUI erstellt. Bei vielen Elektroden ist die Darstellung unübersichtlicher als bei weniger Elektroden. Es ist ziemlich komplex Buffer-BCI zu erweitern, da alleine für die Darstellung der relevanten Signale 4 Funktionen selber geschrieben und 3 bestehende Funktionen ergänzt werden mussten.

7.5 Praktikumsversuch

Das ursprüngliche Ziel war es, den Studierenden der Medizinischen Informatik das Konzept eines BCIs näher zu bringen. Hierzu wurde eine Versuchsanleitung erstellt. Mit

ihr lässt sich das P300-Konzept anhand der matrixSpeller-Anwendung verdeutlichen. Der Versuch kann als Ergänzung bzw. Aufbauversuch zu dem bisherigen EEG-Versuch oder als eigenständige Aufgabe genutzt werden. Schön wäre es hier, wenn später noch die anderen Konzepte wie SSVEP und IM genutzt werden könnten. Der Versuch ist an einem Vormittag gut durchführbar und bietet zeitlich gesehen noch Platz für die anderen Konzepte.

7.6 Ausblick

Geht man auf die Internetseite, die das Framework Buffer-BCI zur Verfügung stellt, sieht man, dass Buffer-BCI immer noch weiterentwickelt wird. Somit könnte es noch dazu kommen, dass auch Experimente mit SSVEP und IM durchgeführt werden können. Dann kann auch die Versuchsanleitung für den Praktikumsversuch um SSVEP- und IM-Experimente erweitert werden.

8 Literaturverzeichnis

[1] H. Berger, *Über das Elektrenkephalogramm des Menschen*. 1929.

[2] W. Pschyrembel, *Pschyrembel Wörterbuch Pflege*. Walter de Gruyter, 2003.

[3] I. Wellach, *Praxisbuch EEG: Grundlagen, Befundung, Beurteilung und differenzialdiagnostische Abgrenzung*. Georg Thieme Verlag, 28.01.2015. Kapitel 6 und 7.

[4] J. N. Helmut Buchner, *Evozierte Potenziale, Neurovegetative Diagnostik, Okulographie*. 2005.

[5] Universität Trier, "Ereigniskorrelierte potentiale - eine kurze einführung." `http://www.neurolabor.de/ereigniskorreliert.pdf`, 21.01.2003. Letzter Zugriff: 23.08.2015.

[6] Jochen Laubrock, "Matlab & psychtoolbox intro." http://141.89.49.17/ jochen/manuals/PTB-Intro.pdf, 06.01.2013. Letzter Zugriff: 23.08.2015.

[7] MathWorks, "Werbung : Neue ideen erforschen." `http://de.mathworks.com/products/matlab/?refresh=true`. Letzter Zugriff: 21.08.2015.

[8] M. Oehler, "Kapazitive elektroden zur messung bioelektrischer signale," 15.06.2009.

[9] "Toyota develops mind-controlled wheelchair." `https://github.com/jadref/buffer_bci/blob/master/README.md`. – Letzter Zugriff: 19.08.2015.

[10] K. R. P. David Rotermund, Udo A. Ernst, "Toward high performance, weakly invasive brain computer interfaces using selective visual attention," *The Journal of Neuroscience*, 03.04.2013. doi: 10.1523/JNEUROSCI.4225-12.2013.

[11] M. T. Johannes Höhne, Klaus-Robert Müller, "Motor imagery for severely motor-impaired patients: Evidence for brain - computer - interfacing as superior control solution," *PlosOne*, 27.08.2014. doi:10.1371/journal.pone.0104854.

[12] S.-W. L. No-Sang Kwak, Klaus-Robert Müller, "A lower limb exoskeleton control system based on steady state visual evoked potentials," *Journal of Neural Engineering*, 17.08.2015. doi:10.1088/1741-2560/12/5/056009.

[13] Jason Farquhar, "Buffer - bci readme.md." `https://github.com/jadref/buffer_bci/blob/master/README.md`. – Letzter Zugriff: 20.08.2015.

Anhang

A Versuchsanleitung

Praktikum Medizinische Signal – und Bildverarbeitung

P300 – Speller – Experiment

Dokumenteneigenschaften

Autor	Beatrice Baumann
Erstellung	25.09.2015
Überarbeitet am	
Revisionsnummer	V01-001
Sprache	deutsch
Dokumentenstatus	Freigegeben

Dokumenten – History

Revision	Status	Datum	Änderungsgrund
V01-001	Erstellung	25.09.2015	

Inhaltsverzeichnis

Vorbereitende Fragen

1) Beschreiben Sie kurz die Signalentstehung des EEG. Welche Zellen des Gehirns und welche Phänomene an diesen Zellen sind hierfür hauptsächlich verantwortlich?
2) In welcher Größenordnung liegen die Potentiale, die beim EEG gemessen werden?
3) Welche BCI-Paradigmen gibt es neben der P300 noch?
4) Was wird über die LED´s an den Elektroden des actiCap-Systems signalisiert?
5) An welchen Arealen des Schädels hat die P300 ihr Maximum?
6) Erklären Sie das Oddball-Paradigma.

1. Aufgabe Hardware aufbauen

Einführung

Die im Labor verfügbare und für diesen Versuch verwendete Hardware besteht aus dem actiCap-System und dem TMSi-Verstärker. Die Hardware wird im Folgenden vorgestellt:

actiCap-System

Das actiCap-System ist ein System zur Signalaufzeichnung von Gehirnwellen. Zu diesem System gehören die actiCap, Elektroden, Controlbox, Splitterbox und Adapter zu verschiedenen Verstärkern. Die einzelnen Komponenten des Systems werden im folgenden Abschnitt beschrieben.

Abbildung 1: actiCap-System

actiCap

Die actiCap wird wie eine Badekappe angezogen. An ihr sind Halterungen befestigt. In diese werden die Elektroden eingesetzt und mit Elektrolyt Gel befüllt, um eine bessere Verbindung zur Kopfhaut, beziehungsweise eine niedrige Impedanz, zu bekommen. Die Halterungen sind nach dem 10-20-System auf der actiCap angebracht.

Abbildung 2: actiCap

Elektroden

Bei den Elektroden handelt es sich um aktive Elektroden. Das bedeutet zum einen, dass direkt in der Elektrode bereits ein Verstärker integriert ist, um eine Verbesse- rung des Signalrauschens zu erreichen, zum anderen kann bei diesen Elektroden eine Impedanzüberwachung realisiert werden.[8]

- Elektroden 1 - 32 (Datenelektroden) Die Elektroden werden über die Splitterbox mit der ControlBox verbunden.

- Groundelektrode (GND) und Referenzelektrode (REF) Die GND und REF müssen an der ControlBox angeschlossen sein, da sonst das Elektrodensystem nicht funktioniert.

Abbildung 3: GND-Elektrode

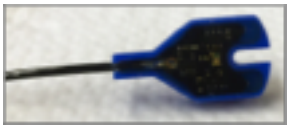


Abbildung 4: REF-Elektrode

Abbildung 5: Datenelektrode

Controlbox

(Im Labor: Controlbox Version II) An der Controlbox sind die Splitterbox und der actiCap-Adapter angeschlossen. Die Splitterbox wird an den Anschluss Ch. 1-32 Splitter- box angeschlossen, der actiCap-Adapter an den Anschluss Ch. 1-32 Amplifier. Außerdem werden mit ihr auch die GND und REF verbunden.

- **Akquisitionsmodus** (Einschalt-Symbol) In diesem Modus können die EEG-Signale zum Verstärker übertragen und aufgezeichnet werden.
- **Impedanzmodus** (Z) In diesem Modus können die Impedanzen der Elektroden ge- messen werden. In den Elektroden sind LEDs eingebaut, die folgende Impedanzen anzeigen:
    - grüne LED: Impedanz unter 25 kΩ
    - gelbe LED: Impedanz zwischen 25 und 60 kΩ
    - rote LED: Impedanz größer als 60 kΩ

Um eine hervorragende Datenqualität während der EEG-Aufzeichnung zu bekommen, ist bei einer Verwendung von aktiven Elektroden eine Impedanz von 25 kΩ vollkommen ausreichend.

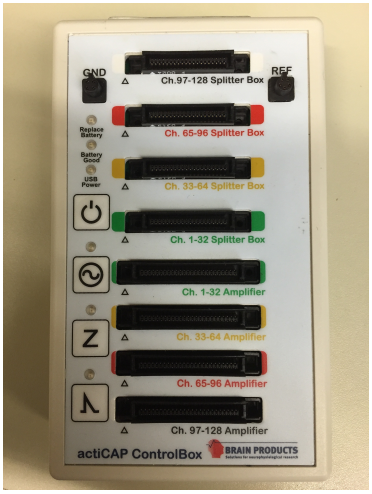


Abbildung 6: Controlbox

Splitterbox

An der Splitterbox sind alle Elektroden angeschlossen (Ausnahme: GND und REF). Die Splitterbox wird mit der Controlbox verbunden.

Abbildung 7: Splitterbox

Adapter

Es gibt verschiedene Adapter für verschiedene Verstärker. Im Labor ist ein Adapter für den TMSi-Verstärker vorhanden. Dieser ist nötig, damit der TMSI-Verstärker mit dem actiCap-System interagieren kann. Bei diesem Adapter kann jede Elektrode einzeln ein- oder ausgeschaltet werden.

Abbildung 8: actiCap-Adapter zu TMSi-Verstärker

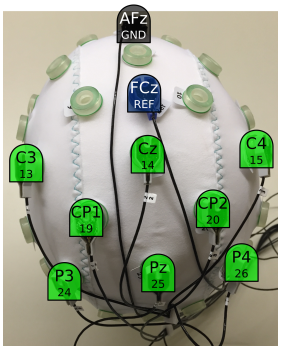
TMSi-Verstärker

(Im Labor: TMSi Refa 8-32e4b4a) Der TMSi-Verstärker wird mit einem Glasfaserkabel über eine Glasfaser zu USB Schnittstelle (FUSBI) direkt mit dem Computer verbunden. Unter anderem besitzt der Verstärker 32 unipolare EEG-Kanäle, 4 weitere bipolare Eingänge sowie einen 8-Bit Trigger-Eingang. Die maximale Abtastrate des Verstärkers beträgt 2048 Hertz. Er wird über den actiCap-Adapter in das actiCap-System integriert.

Abbildung 9: TMSi-Verstärker

Versuchsaufbau

1) Der actiCap-Adapter wird über das breitere Kabel mit dem TMSi-Verstärker verbunden.
2) Der actiCap-Adapter wird über das dünnere Kabel mit der Controlbox an Ch. 1-32 Amplifier angeschlossen.
3) Die Splitterbox wird bei der Controlbox an Ch. 1-32 Splitterbox angeschlossen.
4) Die GND- und REF-Elektrode werden an der Controlbox angeschlossen (GND links und REF rechts)
5) Einer von der Gruppe muss nun die actiCap anziehen. Dazu sollte zuerst der Kopfumfang gemessen werden, um eine passende actiCap anziehen zu können. Im Labor sind 3 Größen vorhanden: für 54 cm, 56 cm und 58 cm Kopfumfang
6) Die Elektroden werden wie in Abbildung 10 & Tabelle 1 gezeigt, an die actiCap angebracht.
7) Den Impedanzmodus einschalten. Hierzu auf der Controlbox den Knopf mit dem Z drücken.
   HINWEIS: Nach einer Weile schaltet sich der Impedanzmodus selbstständig aus. Dieser kann über erneutes drücken von Z wieder eingeschaltet werden.
8) Die Elektroden werden mit Hilfe der Spritze mit Elektrolyt Gel befüllt, bis alle grün leuchten. Hierbei sollte mit der GND- und REF-Elektrode begonnen werden!



Elektrode	Halterung
GND	AF_Z (GND)
REF	FC_Z (REF)
1	C3 (13)
2	C_Z (14)
3	C4 (15)
4	P4 (26)
5	P_Z (25)
6	P3 (24)
7	CP1 (19)
8	CP2 (20)

Abbildung 10 & Tabelle 1: Zuweisung der Elektroden auf der actiCap

Vergewissern Sie sich, dass alle Komponenten gemäß Abbildung 11 angeschlossen sind.

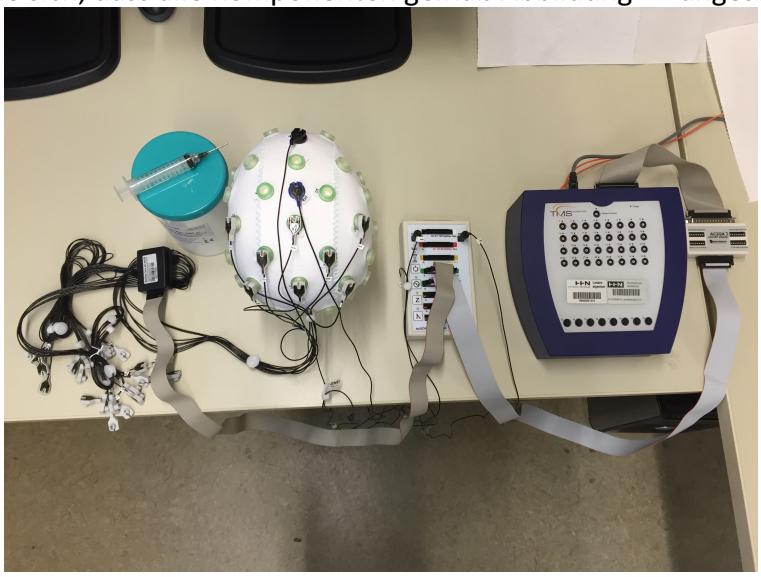


Abbildung 11: Versuchsaufbau

2. Aufgabe EEG-Signale darstellen

Plausibilitätskontrolle

Nachdem eine Person der Gruppe verkabelt ist, gilt es nun die Signale auf dem Computer darzustellen und auf Plausibilität zu überprüfen. Bei der Kontrolle sollte überprüft werden, ob Elektroden störende Signale liefern. Dazu wird zu erst die auf dem Desktop befindliche Datei startTMSI Fieldtrip Buffer Refa ausgeführt. Diese startet den Buffer.
Danach muss die Datei BufferViewer geöffnet werden. Es öffnet sich eine GUI. Die Bufferadresse ist: localhost:1972. Um Signale anzeigen zu können, sollte der Hochpassfilter eingeschaltet werden und anschließend die Verbindung zum Buffer über Connect hergestellt werden. Die Signale sollten nun dargestellt werden.

Anhand verschiedener Bewegungen lässt sich überprüfen, ob die angezeigten Signale plausibel sind.

Zähne knirschen / fest zubeißen

Es sollten deutliche Artefakte in dem Signal zu sehen sein.

„Headshaking“

Dabei sollte die Person, die die actiCap angezogen hat kräftig mit dem Kopf wackeln. Auch hierbei entstehen deutliche Artefakte im Signal.

Augen zu und wieder auf

Die Augen sollten für einen Moment geschlossen werden, dabei entsteht der sogenannte Alpha-Rhythmus. Macht man die Augen wieder auf kommt es zum sogenannten Beta-Rhythmus. Diese Änderung sollte sichtbar sein.

3. Aufgabe P300-Speller-Experiment

Einführung

Die P300 ist eine positive Welle im EEG die als Antwort auf einen spezifischen Reiz gemessen werden kann und ca. 300 Millisekunden nach Reizbeginn auftritt. Sie lässt sich in die P3a und die P3b unterteilen, wobei die P3b der P300 entspricht. Es handelt sich bei der P300 grundsätzlich um ein reizsynchron gemitteltes evoziertes Potential, wobei es zwei verschiedene Reizklassen gibt von welchen der eine seltener ist. Dieser seltenere Reiz wird von der P300-Welle begleitet, der deutlich häufigere Reiz nicht.

Versuchsdurchführung

Die matrixSpeller-Anwendung wird folgendermaßen gestartet:

1. Den Echtzeitdatenbuffer starten über TMSI Fieldtrip Buffer Refa (auf Desktop)
2. Signalanalyseprozess starten über buffer_bci-master/matrixSpeller/startSigProcBuffer
   HINWEIS: warten, bis der Auswahldialog für eine Cap erscheint.
3. Cap auswählen (am besten: cap_tmsi_mobita_p300.txt)
4. Anwendung starten über buffer_bci-master/matrixSpeller/runSpeller

Erklärung der Hauptgui

- CapFitting Bei CapFitting öffnet sich eine GUI, in der man die angeschlossenen Elektroden angezeigt bekommt, hier sollten möglichst alle Elektroden grün sein, was bedeutet, dass die Elektroden-Impedanz niedrig genug ist um eine gute Signalaufzeichnung zu bekommen.
- EEG Hier werden die aktuell aufgezeichneten Signale angezeigt mit dem Sinn, dass man überprüfen kann (z.B. anhand von Augenbewegungen oder Muskelan- spannung (Zähne knirschen)) ob die Signale plausibel sind.
- Practice In Practice kann man, wenn man möchte, üben bevor man den Klassi- fikator kalibriert. Dieser Schritt ist optional.
- Calibration Hiermit wird eine Lernstichprobe erfasst, mit deren Hilfe der Klas- sifikator anschließend trainiert wird. Dem Benutzer wird die Buchstabenmatrix angezeigt. Er muss sich für die Erfassung der Lernstichprobe auf den angezeigten Buchstaben konzentrieren, bis die Erfassung beendet ist. Es ist hilfreich, wenn der Benutzer jedes mal, wenn der Buchstabe aufleuchtet, “ja“ denkt.
- Show Responses Show Responses wurde selber hinzugefügt, um nach der Kalibrierung relevante Signalabschnitte, wie zum Beispiel die gemittelten Reize, anschauen zu können.
- Train Classifier Trainiert den Klassifikator für die Anwendungen.
- Copy Spelling 1 & 2 Hier wird ein Wort vorgegeben, welches man nachbuchsta- bieren soll. Leider wurde noch nicht herausgefunden, wo dieses Wort vorgegeben werden kann.

- Freespelling Nachdem ein Klassifikator kalibriert und trainiert wurde, kann man beim Freespelling Wörter buchstabieren. Hierzu sucht sich der Benutzer einen Buchstaben heraus und konzentriert sich auf diesen. Auch hier ist es hilfreich, jedes mal, wenn der Buchstabe leuchtet, “ja“ zu denken. Freespelling ist standardmäßig so eingestellt, dass nach 5 Buchstaben die Anwendung zu Ende ist.
- ContFeedback ist prinzipiell das Selbe, wie das Freespelling. Mit dem Unter- schied, dass hier nicht nach 5 Buchstaben aufgehört wird.

Aufgabe

Versuchen Sie bei unterschiedlichen Wiederholraten ein Wort mit 5 Buchstaben zu buchstabieren. Notieren Sie ihre Ergebnisse in Tabelle 2: z.B. das Wort „HALLO“ und bei Ergebnis wie viele Buchstaben von den 5 erkannt wurden (Beispiel 5/5)

Wiederholrate einstellen

Um die Wiederholrate einzustellen wird in der configureSpeller-Datei die Variable nRepetitions den entsprechenden Werten angepasst.

Wiederholrate						Ergebnis
5						
2						
1						
0.5						
0.25						

Tabelle 2: Ergebnisse des Versuchs

Fragestellungen

Lässt sich etwas über die Wiederholrate bzw. die Genauigkeit aussagen? Gibt es einen Idealwert? Bedenken Sie auch, die Anstrengung, die sie je Wort benötigt haben. Lassen sich daraus auch Grenzen für das System formulieren?

Spiele

Es werden nach dem gleichen Prinzip auch Spiele (Snake, Pacman und Sokoban) zur Verfügung gestellt.

Die games-Anwendung wird folgendermaßen gestartet:

1. Den Echtzeitdatenbuffer starten über TMSI Fieldtrip Buffer Refa (auf Desktop)
2. Signalanalyseprozess starten über buff_bci-master/games/startSigProcBuffer
   HINWEIS: warten, bis der Auswahldialog für eine Cap erscheint
3. Cap auswählen (am besten: cap_tmsi_mobita_p300.txt)
4. Anwendung starten über buffer_bci-master/games/runGames

Die Bewegungen während des Spiels finden über Pfeile, die unterschiedlich aufleuchten, statt. Der Benutzer konzentriert sich auf einen Pfeil. Immer, wenn dieser Pfeil aufleuchtet, kommt es zu einer P300, anhand derer das BCI sich für diesen Pfeil, und die entsprechende Bewegung des Pfeils, entscheidet. Auch hier ist es hilfreich, wenn man bei jedem Aufleuchten “ja“ denkt.

Aufgabe

Testen Sie eines der Spiele, solange Sie Lust haben.

Denken Sie, dass man anhand dieses Prinzips weitere Anwendungen ermöglichen kann? Welche Anwendungen würden sich hierüber realisieren lassen?

Gibt es schon Umsetzungen, die dieses Paradigma nutzen?

Wenn Sie noch Zeit haben, können sie durchtauschen und die andere Person an das EEG anschließen.

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

(Ort, Datum) (Unterschrift)